UNIVERSITÀ POLITECNICA DELLE MARCHE
FACOLTÀ DI INGEGNERIA
Dipartimento di Ingegneria dell'Informazione
Corso di Laurea in Ingegneria Informatica e dell'Automazione

TESI DI LAUREA

Esperienze nel contesto dell'Intelligenza Artificiale: Chatbot, Riconoscimento di Immagini, Sentiment Analysis

Experiences in the context of Artificial Intelligence: Chatbot, Image Recognition, Sentiment Analysis

Relatore

Prof. Domenico Ursino

Candidato

Mattia Mandorlini

ANNO ACCADEMICO 2023-2024

"La vera domanda non è se le macchine penseranno, ma se le persone penseranno. La sfida non è replicare l'intelligenza, ma comprendere l'intelligenza umana."

George Dyson

Sommario

L'interesse per l'Intelligenza Artificiale sta crescendo in tutto il mondo, diventando un argomento comune di discussione in molteplici contesti; ma l'IA non è una novità: da circa un decennio miliardi di persone la utilizzano quotidianamente. Attività come la ricerca su Google o l'utilizzo di Maps per la pianificazione di viaggi sono ormai abitudini consolidate, rese possibili proprio grazie all'IA. Queste applicazioni contribuiscono a migliorare la comunicazione, i trasporti, la sicurezza e la salute delle persone, senza richiedere sforzi particolari da parte degli utenti. Tuttavia, nonostante la facilità d'uso sia uno dei vantaggi principali dell'IA, è essenziale che le persone comprendano come e con quali scopi questa tecnologia venga sviluppata. L'obiettivo di questa tesi è quello di introdurre il concetto di Intelligenza Artificiale partendo da concetti preliminari e dalla storia fino ad arrivare ad analizzare i rischi e i benefici ad essa associati. Si passerà, poi, allo studio di uno dei pilastri del cloud computing, ovvero Amazon Web Service, e soltanto alla fine esploreremo nel dettaglio tre tra le tante applicazioni dell'Intelligenza Artificiale.

Keyword: Intelligenza Artificiale (IA), Machine Learning (ML), Deep Learning (DL), Big data, Reti neurali, Data Science, Cloud Computing

Indice

Elenco delle figure

Introduzione

L'Intelligenza Artificiale (IA), definibile come la capacità di un sistema informatico di eseguire compiti che tipicamente vengono svolti dall'intelligenza umana, permette alle macchine di apprendere dai dati, trarre conclusioni, adattarsi alle circostanze e compiere azioni intelligenti, il tutto senza un intervento umano diretto. Questa definizione, per quanto semplice, cela un background di complessità e potenziale e ci rivela come l'IA sia una forza trainante di trasformazione nel panorama moderno, permeando ogni settore, dalle applicazioni quotidiane alle industrie avanzate. Tuttavia, la sua complessità non può essere sottovalutata; gli algoritmi complessi possono sollevare dubbi sulla trasparenza e l'equità delle decisioni, mentre il rischio di distorsioni e discriminazione nei dati e le preoccupazioni sulla privacy e sicurezza richiedono un approccio attento e responsabile. In questo delicato equilibrio tra promesse e sfide, è fondamentale perseguire un utilizzo etico e responsabile dell'IA per massimizzare i benefici e mitigare i rischi nell'era digitale in cui ci troviamo immersi.

La presente tesi è nata per l'interesse personale nell'esplorare le profondità di questa tecnologia rivoluzionaria e comprendere appieno il suo potenziale e le sue implicazioni. Inoltre, l'IA rappresenta un campo di studio multidisciplinare che interseca informatica, matematica, psicologia e filosofia, offrendo un terreno fertile per la ricerca e l'innovazione. Tuttavia, al di là della mera curiosità accademica, vi è anche una motivazione pratica e applicativa. In un mondo sempre più guidato dall'IA, è cruciale comprendere come questa tecnologia possa essere utilizzata in modo efficace per risolvere problemi complessi e migliorare la qualità della vita umana. Allo stesso tempo, è altrettanto importante riconoscere e affrontare le sfide e le implicazioni negative che potrebbero sorgere lungo il cammino.

All'interno della presente tesi sono stati realizzati e analizzati tre case study inerenti ad alcune applicazioni dell'Intelligenza Artificiale, il tutto mediante il servizio cloud di Amazon, ovvero Amazon Web Service (AWS), di cui, tra le sue molteplici offerte, spicca l'ampia gamma di servizi dedicati all'IA, con particolare enfasi per ciò che concerne il machine learning.

Le esperienze che si realizzeranno riguardano tematiche ormai molto diffuse, tra cui i chatbot, il riconoscimento facciale e la sentiment analysis. Non è esagerato affermare che, al giorno d'oggi, ognuno di noi, seppur inconsapevolmente, si imbatte più volte al giorno con queste tecnologie.

Nella realizzazione del chatbot verrà innanzitutto selezionato un contesto applicativo, in cui il suo utilizzo potrà essere di notevole aiuto. L'ambito fitness, al giorno d'oggi, nasconde molti punti di dibattito; pertanto, sarà fornito un assistente virtuale intelligente per aiutare le persone a chiarire i loro dubbi. In termini tecnici, prima di iniziare la realizzazione effettiva del chatbot, verrà creato un *conversation flow diagram* per organizzarne la struttura e poterne

prevedere, per quanto possibile, alcuni flussi di conversazione. Successivamente, si procederà alla configurazione degli intent tramite la console di Amazon Lex, dove si definiranno e gestiranno le azioni che il chatbot dovrà eseguire. Per ciascun intent si addestrerà il modello affinché sia in grado di interpretare correttamente le richieste degli utenti e di agire di conseguenza. Tale processo comporterà la definizione di varie frasi di esempio (utterance), l'identificazione di variabili di contesto (slot) e l'implementazione di funzioni Lambda. Terminata la progettazione sulla console, si procederà alla distribuzione del chatbot completo sulla piattaforma Telegram. Per realizzare ciò saranno impiegati altri servizi AWS tra cui Amazon API Gateway e Amazon DynamoDB. Verrà, quindi, creata un'API per consentire la comunicazione tra il servizio Amazon Lex e il chatbot di Telegram; verrà inoltre realizzata una funzione Lambda per la memorizzazione di informazioni su un database.

Il secondo case study prevederà l'applicazione della Computer Vision nel contesto del riconoscimento facciale. Si progetterà un'architettura più complessa che coinvolgerà diversi servizi AWS in interazione con un'applicazione front-end sviluppata utilizzando la libreria *React JS*. Inizialmente, si procederà con la progettazione e il completamento dell'interfaccia grafica, per poi passare all'organizzazione del back-end e dell'API, fino alla progettazione dello storage dei dati. Successivamente, verranno sviluppate le funzioni per la raccolta e la configurazione dei dati lato client e quelle per la gestione e l'elaborazione dei dati lato server. Dopo di ciò, si implementerà e testerà l'API utilizzando Amazon API Gateway. Una volta verificata la correttezza delle connessioni e lo scambio di dati, si procederà con le connessioni ai servizi S3 e DynamoDB per il salvataggio dei dati.

Infine, con l'ultimo case study, verranno testate le potenzialità e le utilità della sentiment analysis. L'obiettivo di questo progetto sarà quello di valutare la coerenza tra il testo di una recensione e il numero di stelle assegnate. Utilizzando i dati ottenuti dall'analisi del testo tramite IA, si proporrà una valutazione in stelle del testo stesso, insieme ai livelli di sentimento associati alle diverse entità presenti nella recensione. L'architettura del progetto sarà simile al caso precedente, ma senza alcune complessità poiché non sarà necessario memorizzare alcun dato. Si seguirà un approccio analogo al caso precedente, iniziando con la progettazione e l'implementazione dell'interfaccia grafica, per poi procedere con lo sviluppo delle funzionalità lato client e lato server, terminando, infine, con la realizzazione e il test delle API.

La presente tesi sarà composta da sei capitoli strutturati come di seguito specificato:

- Nel Capitolo 1 si introdurrà il concetto di Intelligenza Artificiale e verranno discussi i contributi che diverse discipline hanno apportato alla sua creazione e sviluppo; in seguito, si approfondirà la storia dell'IA fino ai giorni nostri e, conseguentemente, si parlerà dei vari tipi di IA e dei modelli di apprendimento sviluppati, fino ad arrivare ad una discussione riguardante i rischi e i benefici che queste tecnologie portano all'essere umano.

- Nel Capitolo 2 si presenterà Amazon Web Service (AWS), un servizio di cloud computing. In particolare, verrà trattato brevemente il concetto di cloud per poi andare nel dettaglio di AWS con la storia e l'evoluzione, l'architettura globale e, infine, l'analisi dei servizi utilizzati.

- Nel Capitolo 3 si approfondirà il concetto di "Chatbot" partendo con un'introduzione all'argomento fino ad arrivare ad una riflessione sui rischi derivanti dal loro utilizzo e all'attuale evoluzione di questa tecnologia. Si passerà, poi, ad analizzare dettagliatamente l'esperienza realizzata.

- Nel Capitolo 4 si esplorerà il concetto di "Computer Vision", analizzando le tecnologie fondamentali che la supportano, tra cui il deep learning e le reti neurali; successivamen-

te, si esploreranno le sue applicazioni pratiche. Si approfondirà poi, un'applicazione specifica della computer vision, ossia il riconoscimento di immagini, di cui si esamineranno le fasi chiave del suo funzionamento, si discuteranno i possibili sviluppi futuri e le applicazioni emergenti. Infine, si descriverà in dettaglio l'esperienza realizzata partendo dal front-end fino ad arrivare al back-end e ai servizi di storage.

- Nel Capitolo 5 si esaminerà un'altra applicazione dell'Intelligenza Artificiale, ovvero la sentiment analysis. Si partirà con un'introduzione al concetto fino ad arrivare ad aspetti più tecnici come il funzionamento e le varie tipologie. Successivamente, faremo una riflessione sulla sua importanza e sulle sfide che comporta. A terminare il capitolo vi sarà la descrizione dell'ultima esperienza realizzata che, come nel capitolo precedente, sarà trattata nel dettaglio descrivendo tutta l'architettura.
- Nel Capitolo 6 verranno esposte alcune riflessioni e considerazioni in merito al lavoro svolto e, per ogni esperienza, verranno tratte delle conclusioni riguardanti i limiti e le potenzialità riscontrati nell'uso delle varie tecnologie.

CAPITOLO 1

Introduzione all'Intelligenza Artificiale

In questo capitolo esploreremo il concetto di "Intelligenza Artificiale" (IA), partendo da concetti fondamentali come la razionalità umana e il pensiero. Successivamente, esamineremo i contributi delle varie discipline all'IA e tracceremo il percorso storico che ha condotto alla sua nascita e alla sua evoluzione fino ad oggi. Discuteremo le diverse tipologie di IA e i modelli di apprendimento sviluppati. Infine, affronteremo i rischi e i benefici di questa innovazione.

1.1 Cos'è l'Intelligenza Artificiale

Ci definiamo Homo sapiens - l'uomo saggio - poiché l'intelligenza riveste per noi un'importanza fondamentale. Da millenni cerchiamo di comprendere il funzionamento del nostro pensiero e delle nostre azioni, ossia come il nostro cervello riesca a percepire, comprendere, prevedere e manipolare un mondo molto più vasto e complesso di se stesso. Il campo dell'Intelligenza Artificiale, o IA, non si limita solo alla comprensione, ma mira anche a creare entità intelligenti, ovvero macchine capaci di calcolare come agire in modo efficace e sicuro in una vasta gamma di situazioni nuove.

Nel corso del tempo i ricercatori hanno fornito varie definizioni di IA; alcuni hanno delineato l'intelligenza in termini di quanto sia vicina alle prestazioni umane, mentre altri hanno preferito una definizione astratta e formale dell'intelligenza nota come razionalità, che, in linea generale, significa compiere la "cosa giusta". Inoltre, il concetto stesso di razionalità può variare: alcuni la considerano come una caratteristica dei processi di pensiero interni e del ragionamento, mentre altri focalizzano l'attenzione sul comportamento intelligente, che rappresenta una valutazione esterna. A partire da queste due dimensioni - umano vs. razionale e pensiero vs. comportamento - emergono quattro combinazioni possibili, ciascuna delle quali ha avuto sostenitori e programmi di ricerca dedicati.

1.1.1 Agire in modo umano: l'approccio del test di Turing

Il test di Turing, proposto da Alan Turing (1950), è stato ideato come un esperimento mentale che avrebbe evitato la vaghezza della domanda "Può una macchina pensare?" Un computer supera il test se un interrogatore umano, dopo aver posto alcune domande scritte, non riesce a distinguere se le risposte scritte provengono da una persona o da un computer. (Figura 1.1). Il computer per passare il test avrebbe bisogno delle seguenti capacità:

- *elaborazione del linguaggio naturale*, per comunicare con successo in una lingua umana;

- *rappresentazione della conoscenza*, per memorizzare ciò che sa o sente;
- *ragionamento automatizzato*, per rispondere alle domande e trarre nuove conclusioni;
- *apprendimento automatico*, per adattarsi a nuove circostanze e per rilevare ed estrapolare modelli.

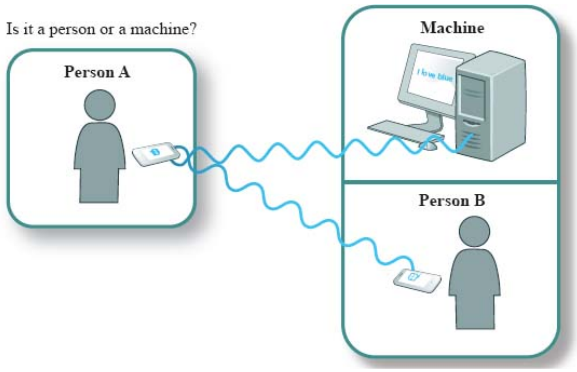


Figura 1.1: Immagine di esempio di un test di turing

Turing riteneva che simulare fisicamente una persona fosse superfluo per dimostrare l'intelligenza. Tuttavia, altri ricercatori hanno proposto un test di Turing totale, che richiede l'interazione con oggetti e persone nel mondo reale. Per superare questo test, un robot dovrà essere dotato di *visione artificiale* e *riconoscimento vocale*, per percepire l'ambiente, nonché di *abilità robotiche*, per manipolare gli oggetti e spostarsi. Queste sei discipline costituiscono la base dell'IA.

1.1.2 Pensare umanamente: l'approccio del modellamento cognitivo

Per dire che un programma pensa come un essere umano, dobbiamo conoscere il modo in cui gli esseri umani pensano. Possiamo apprendere il pensiero umano in tre modi:

1. *Introspezione:* osservando i nostri stessi pensieri mentre passano;
2. *Esperimenti psicologici*: osservando una persona in azione;
3. *Imaging cerebrale*: osservando il cervello in azione.

Una volta che abbiamo una teoria della mente sufficientemente precisa, diventa possibile esprimerla come un programma per computer. Se il comportamento input-output del programma coincide al comportamento umano corrispondente, ciò è prova che alcuni dei meccanismi del programma potrebbero essere presenti anche negli esseri umani.

Il campo interdisciplinare della scienza cognitiva unisce modelli informatici provenienti dall'IA e tecniche sperimentali dalla psicologia per costruire teorie precise e verificabili della mente umana. Un esempio lampante lo abbiamo soprattutto nella visione artificiale, che integra evidenze neurofisiologiche nei modelli computazionali. Di recente, l'unione di metodologie di neuroimaging e tecniche di apprendimento automatico per l'analisi dei dati ha avviato un processo di sviluppo della capacità di "leggere le menti", ossia di comprendere il contenuto semantico dei pensieri interni di un individuo.

1.1.3 Pensare razionalmente: l'approccio delle "leggi del pensiero"

Il filosofo greco Aristotele fu tra i pionieri nel tentativo di codificare il "pensiero corretto", ossia i processi di ragionamento che conducono a conclusioni irrefutabili. I suoi sillogismi fornirono modelli di argomentazione che sempre portavano a conclusioni corrette quando basate su premesse corrette. Un esempio classico è "Socrate è un uomo" e "tutti gli uomini sono mortali", che conduce alla conclusione che "Socrate è mortale".

Si credeva che queste leggi del pensiero governassero il funzionamento della mente, e lo studio di esse diede origine al campo della logica. Tuttavia, la logica, nella sua accezione più comune, richiede una conoscenza certa del mondo - una condizione raramente raggiunta nella realtà. Le regole del mondo sono spesso sconosciute o incerte. La teoria della probabilità colma questa lacuna consentendo un ragionamento rigoroso anche con informazioni incerte. Essa offre la possibilità di costruire un modello completo del pensiero razionale partendo dalle informazioni percettive grezze fino alla comprensione del funzionamento del mondo e alle previsioni sul futuro.

Tuttavia, la teoria della probabilità non è in grado di generare comportamenti intelligenti. Per questo abbiamo bisogno di una teoria dell'azione razionale. Il pensiero razionale, pur essendo fondamentale, da solo non è sufficiente per generare comportamenti intelligenti.

1.1.4 Agire razionalmente: l'approccio dell'agente razionale

Un agente razionale è qualcosa che agisce in autonomia, percependo l'ambiente, adattandosi nel tempo e perseguendo obiettivi specifici. L'obiettivo di un agente razionale è ottenere il miglior risultato possibile, o almeno il migliore risultato atteso in presenza di incertezza.

Nell'approccio tradizionale dell'IA, l'attenzione era sul fare inferenze corrette; tuttavia, agire razionalmente può implicare anche comportamenti che non coinvolgono necessariamente l'inferenza, come reazioni riflessive.

Tutte le abilità richieste per superare il test di Turing permettono anche agli agenti di agire razionalmente. La rappresentazione della conoscenza e il ragionamento sono fondamentali per prendere decisioni informate, mentre l'apprendimento migliora l'efficacia del comportamento, soprattutto in situazioni nuove.

L'approccio dell'agente razionale all'IA ha il vantaggio di essere più generale e scientificamente sviluppabile rispetto ad altri approcci. Il suo obiettivo è quello di costruire agenti che compiano azioni razionali, il cui concetto di "razionale" è definito dagli obiettivi che forniamo all'agente.

Anche se la razionalità perfetta è raramente realizzabile in ambienti complessi, resta comunque un punto di partenza per l'analisi teorica, anche se spesso deve essere bilanciata con la necessità di agire rapidamente in situazioni reali.

1.2 I fondamenti dell'Intelligenza Artificiale

In questa sezione, esploreremo brevemente le discipline che hanno contribuito con idee, prospettive e tecniche all'Intelligenza Artificiale, organizzando la storia attorno a una serie di domande. (Figura 1.2)

1.2.1 Filosofia

Possono essere utilizzate regole formali per trarre conclusioni valide?
Aristotele (384-322 a.C.) fu il primo a elaborare un preciso insieme di leggi che regolano la parte razionale della mente. Egli sviluppò un sistema informale di sillogismi per il ragionamento corretto che, in teoria, consentiva di generare conclusioni in modo meccanico,

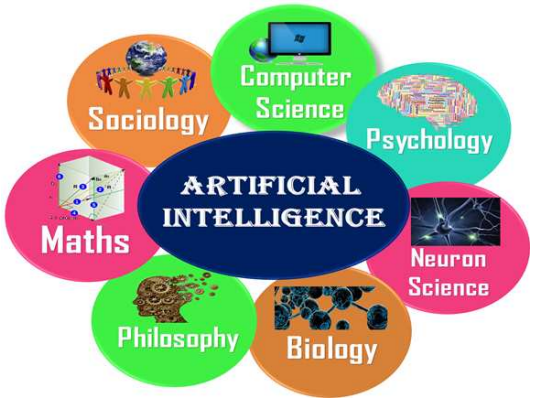


Figura 1.2: Immagine di alcune discipline fondamentali dell'IA

date certe premesse iniziali. Ramon Llull (circa 1232-1315) ideò un sistema con l'intento di formalizzare il ragionamento umano attraverso dispositivi meccanici, tra cui un insieme di ruote di carta che potevano essere ruotate in diverse permutazioni, dimostrando, così, il desiderio di applicare regole formali alla logica.

Nel suo libro "Leviathan" del 1651, Thomas Hobbes (1588-1679) avanzò l'idea di una macchina pensante, un "animale artificiale", come lo definì, argomentando che "il cuore è una molla; i nervi sono cavi; le articolazioni sono ruote." Egli suggerì, inoltre, che il ragionamento fosse simile alla computazione numerica, sostenendo che "ragionare" non fosse altro che "calcolare", ovvero aggiungere e sottrarre.

Tuttavia, una cosa è affermare che la mente operi, almeno in parte, secondo regole logiche o numeriche[1], e costruire sistemi fisici che emulino alcune di queste regole, un'altra cosa è affermare che la mente stessa sia un tale sistema fisico.[2]

Come sorge la mente da un cervello fisico?
René Descartes (1596-1650) fu il primo a delineare chiaramente la distinzione tra mente e materia. Egli notò che una concezione puramente fisica della mente sembrava limitare il concetto di libero arbitrio. Se la mente è interamente soggetta alle leggi fisiche, non ha più libero arbitrio di quanto non ne abbia una roccia nel decidere di cadere verso il basso. Descartes era un sostenitore del dualismo, che sosteneva l'esistenza di una parte della mente umana (o anima o spirito) al di fuori delle leggi fisiche, esente da esse. Al contrario, gli animali non possedevano questa dualità; potevano essere considerati come semplici macchine. Un'alternativa al dualismo è il materialismo, che propone che il funzionamento del cervello secondo le leggi della fisica costituisca la mente. Il concetto di libero arbitrio è, semplicemente, la percezione delle opzioni disponibili per l'individuo che sceglie.

Da dove proviene la conoscenza?
Il passo successivo consiste nel determinare la fonte della conoscenza. Nell'ambito dell'empi-

[1] Questo concetto si riferisce al fatto che il processo decisionale e il ragionamento umano possono essere in parte formalizzati attraverso regole logiche o numeriche. Le menti umane spesso seguono modelli di pensiero razionale che possono essere descritti e compresi attraverso concetti logici e matematici. Ad esempio, quando risolviamo un problema matematico o prendiamo decisioni basate su logica e ragionamento, stiamo operando secondo regole logiche o numeriche.

[2] Questo concetto riguarda l'idea che la mente stessa sia un sistema fisico, cioè che la nostra capacità di pensare, ragionare e prendere decisioni sia, in definitiva, il risultato di processi fisici che avvengono nel nostro cervello e nel nostro corpo. In altre parole, la mente umana è considerata il prodotto di attività cerebrali e biochimiche che si verificano nel nostro sistema nervoso.

rismo, viene enfatizzato il principio di John Locke (1632-1704): "Nulla è nell'intelletto che non sia stato prima nei sensi".

Il Circolo di Vienna (Sigmund, 2017), composto da filosofi e matematici, basandosi sul lavoro di Ludwig Wittgenstein (1889-1951) e Bertrand Russell (1872-1970), sviluppò la dottrina del positivismo logico. Questa teoria sostiene che tutta la conoscenza può essere descritta da teorie logiche collegate, le quali, alla fine, trovano riscontro in frasi di osservazione che corrispondono agli input sensoriali; quindi, il positivismo logico fonde elementi di razionalismo ed empirismo.

La teoria della conferma, elaborata da Rudolf Carnap (1891-1970) e Carl Hempel (1905-1997), tentò di analizzare l'acquisizione di conoscenze dall'esperienza, quantificando il grado di credenza che dovrebbe essere attribuito alle frasi logiche in base al loro legame con osservazioni che le confermano o le smentiscono.

Come la conoscenza porta all'azione?
L'ultima componente nell'immagine filosofica della mente riguarda il legame tra conoscenza e azione. Questo aspetto è cruciale per l'Intelligenza Artificiale poiché l'intelligenza richiede sia l'azione che il ragionamento. Solo comprendendo come le azioni siano giustificate possiamo capire come costruire un agente il cui comportamento sia razionale.

Aristotele, nel suo lavoro "De Motu Animalium", argomentò che le azioni sono giustificate dalla connessione logica tra obiettivi e la conoscenza dell'esito dell'azione. Nell'"Etica Nicomachea", approfondì questo argomento suggerendo un algoritmo secondo cui si delibera sui mezzi, non sugli scopi. Questo porta a considerare come raggiungere un obiettivo e valutare quali mezzi adottare fino a giungere alla causa primaria. L'algoritmo aristotelico è stato implementato 2300 anni dopo da Newell e Simon nel loro programma Risolutore di Problemi Generale.

Pensare solo in termini di azioni che perseguono obiettivi può essere utile ma non sempre applicabile. Ad esempio, quando ci sono diverse vie per raggiungere un obiettivo, occorre scegliere tra di esse. Inoltre, potrebbe non essere possibile raggiungere un obiettivo con certezza, ma è, comunque, necessario agire. In queste situazioni, si deve decidere in base alla massimizzazione dell'utilità attesa.

In materia di etica e politica pubblica, l'approccio utilitaristico propone di massimizzare l'utilità, considerando gli interessi di tutti gli individui coinvolti. Al contrario, l'approccio deontologico di Kant si basa su regole universali che determinano la moralità delle azioni. Molti sistemi di Intelligenza Artificiale moderni adottano un approccio che combina l'utilitarismo con la considerazione dei principi etici fondamentali.

1.2.2 Matematica

Quali sono le regole formali per trarre conclusioni valide?
Le regole formali che guidano l'elaborazione di conclusioni valide si originano dalla formalizzazione matematica della logica e della probabilità. La logica formale, elaborata da figure come George Boole e Gottlob Frege, ha dato vita alla logica proposizionale e alla logica del primo ordine, pilastri fondamentali nel ragionamento informatico e nell'Intelligenza Artificiale. La teoria della probabilità, emersa con Gerolamo Cardano e sviluppata successivamente da pensatori come Blaise Pascal e Pierre Laplace, estende la logica per gestire informazioni incerte, diventando vitale per affrontare misurazioni incerte e teorie incomplete.

Cosa può essere calcolato?
L'informatica, come disciplina formale, si fonda sulla teoria dei numeri e su concetti algoritmici come l'algoritmo di Euclide, il quale costituisce uno dei primi esempi di algoritmo

non banale. Il termine "algoritmo" deriva dall'opera di Muhammad ibn Musa al-Khwarizmi, mentre le discussioni su algoritmi di deduzione logica affondano le radici nel lavoro di Boole. Kurt Gödel, con il suo teorema di incompletezza, ha dimostrato l'esistenza di enunciati veri che non possono essere provati in un sistema formale, rivelando, così, i limiti della deduzione. Inoltre, Alan Turing, ispirandosi al lavoro di Gödel, ha cercato di definire quali funzioni possono essere calcolate da una macchina di Turing, dando vita alla tesi di Church-Turing sull'identificazione della computabilità con funzioni realizzabili da una macchina di Turing.

Come ragioniamo con informazioni incerte?
L'importanza della computabilità e della complessità computazionale, incluso il discernimento tra problemi trattabili e intrattabili, riveste un ruolo cruciale nell'ambito dell'Intelligenza Artificiale. La teoria della NP-completezza[3], introdotta da Cook e Karp, fornisce un fondamento per valutare la trattabilità dei problemi, suggerendo che i problemi NP-completi siano verosimilmente intrattabili. Questi concetti contrastano con l'entusiasmo iniziale verso i computer come risolutori onnipotenti di tutti i problemi, mettendo in evidenza la necessità di una gestione oculata delle risorse e l'accettazione dell'imperfezione necessaria nei sistemi intelligenti.

1.2.3 Economia

Come dovremmo prendere decisioni in base alle nostre preferenze?
Dovremmo basare le nostre decisioni sulle nostre preferenze personali e sui nostri obiettivi. In economia, la teoria delle decisioni combina la teoria della probabilità con la teoria dell'utilità per fornire un quadro formale e completo per le decisioni individuali fatte in condizioni di incertezza. Questo approccio è particolarmente utile per "grandi" economie, in cui gli agenti non devono tener conto delle azioni degli altri agenti come singoli individui.

Come dovremmo procedere quando gli altri potrebbero non essere d'accordo?
In economia, quando ci sono disaccordi tra le parti, è possibile ricorrere alla teoria dei giochi. Quest'ultima, sviluppata da John von Neumann e Oskar Morgenstern, fornisce strumenti per analizzare situazioni in cui le azioni di un giocatore possono influenzare significativamente l'utilità di un altro giocatore, sia in senso positivo che negativo.

Come dovremmo agire quando il "payoff" potrebbe essere ritardato nel tempo?
In questa prospettiva, i problemi decisionali possono essere visti come processi decisionali sequenziali, in cui le azioni prese in sequenza possono influenzare i "payoff[4]" successivi. Un esempio di questa teoria sono i processi decisionali di Markov, formalizzati da Richard Bellman, che trovano applicazioni in operazioni di ricerca e apprendimento per rinforzo. Inoltre, l'approccio della "soddisfazione", introdotto da Herbert Simon, suggerisce che talvolta è più pratico adottare decisioni che siano "abbastanza buone" anziché cercare l'ottimalità assoluta, soprattutto quando le informazioni o i "payoff" futuri sono incerti.

[3] Un problema computazionale è classificato come NP-completo se appartiene alla classe di complessità NP (Non-deterministic Polynomial time) e ha la proprietà che ogni altro problema NP può essere ridotto a esso in tempo polinomiale.

[4] Il "payoff" si riferisce al beneficio o alla ricompensa derivante da una decisione o da un'azione. Quando il "payoff" potrebbe verificarsi in un momento futuro, la teoria delle decisioni sequenziali offre un quadro utile.

1.2.4 Neuroscienza

Come i cervelli elaborano le informazioni ?
La neuroscienza studia il sistema nervoso, in particolare il cervello, e il modo in cui elabora le informazioni è ancora un mistero. Sebbene sia noto che il cervello abilita il pensiero, i dettagli di questo processo sono stati riconosciuti solo recentemente. Paul Broca fu tra i primi a identificare un'area specifica del cervello coinvolta nella produzione del linguaggio, aprendo la strada allo studio dell'organizzazione funzionale cerebrale. Le moderne tecniche di imaging, come la risonanza magnetica funzionale (fMRI), ci permettono di vedere l'attività cerebrale con dettagli senza precedenti. Tuttavia, non capiamo completamente come funzionano le diverse regioni cerebrali o come il cervello gestisca le informazioni incerte. Lo sviluppo delle interfacce cervello-macchina promette di far luce su questi processi complessi, dimostrando la straordinaria capacità del cervello di adattarsi e interagire con il mondo esterno.

I cervelli e i computer digitali presentano differenze sostanziali nelle loro caratteristiche. Sebbene i computer abbiano un tempo di ciclo molto più veloce dei cervelli, con una rapidità un milione di volte maggiore, i cervelli compensano questa disparità con una capacità di archiviazione e interconnessione molto maggiore (Figura 1.3). Anche se i supercomputer più avanzati si avvicinano alle prestazioni cerebrali su alcuni fronti, i futuristi interpretano questi dati come un segnale di avvicinamento a una singolarità, in cui i computer potrebbero raggiungere prestazioni sovrumane e migliorarsi rapidamente. Tuttavia, i confronti puramente numerici non forniscono una comprensione completa della situazione. Anche con un computer di capacità virtualmente illimitata, è essenziale sviluppare ulteriori teorie e concetti per comprendere appieno l'intelligenza. In altre parole, se non si dispone della giusta teoria, una maggiore velocità dei computer può solo portare a una più rapida produzione di risposte sbagliate.

	Supercomputer	Personal Computer	Human Brain
Computational units	10^6 GPUs + CPUs	8 CPU cores	10^6 columns
	10^{15} transistors	10^{10} transistors	10^{11} neurons
Storage units	10^{16} bytes RAM	10^{10} bytes RAM	10^{11} neurons
	10^{17} bytes disk	10^{12} bytes disk	10^{14} synapses
Cycle time	10^{-9} sec	10^{-9} sec	10^{-3} sec
Operations/sec	10^{18}	10^{10}	10^{17}

Figura 1.3: Immagine di comparazione in diversi aspetti tra un supercomputer, un normale PC e il cervello umano

1.2.5 Psicologia

Come pensano e agiscono gli esseri umani e gli animali?
Le origini della psicologia scientifica risalgono al lavoro di Hermann von Helmholtz e Wilhelm Wundt, che applicarono il metodo scientifico allo studio della visione umana e fondarono il primo laboratorio di psicologia sperimentale.

Il comportamentismo, guidato da John Watson, ha rifiutato la teoria dei processi mentali in quanto riteneva che l'introspezione, cioè l'auto-osservazione dei propri processi mentali, non fosse una fonte affidabile di dati scientifici. Invece, il comportamentismo ha preferito concentrarsi sullo studio di misure oggettive dei comportamenti osservabili, come le risposte agli stimoli ambientali, e delle azioni degli animali. Questa prospettiva ha posto l'attenzione su ciò che è esternamente osservabile e misurabile, piuttosto che sulle esperienze interne e soggettive dei soggetti. In sostanza, il comportamentismo ha cercato di rendere la psicologia più scientifica concentrando la sua indagine su comportamenti misurabili e osservabili, anziché su processi mentali difficilmente quantificabili.

La psicologia cognitiva, risalente agli studi di William James, considera il cervello come un dispositivo di elaborazione delle informazioni ed è emersa grazie al lavoro di Frederic Bartlett e Kenneth Craik. Quest'ultimo ha delineato un modello di agente basato sulla conoscenza, sottolineando l'importanza di tradurre stimoli in rappresentazioni interne, manipolare tali rappresentazioni con processi cognitivi e, quindi, ritrasformarle in azioni. Il campo della scienza cognitiva ha preso forma negli Stati Uniti con lo sviluppo della modellazione informatica, dimostrando come i computer potessero essere utilizzati per affrontare questioni di memoria, linguaggio e pensiero logico. L'interazione uomo-computer (HCI) è stata anche considerata sotto il campo della psicologia, con Doug Engelbart, che ha sostenuto l'idea di potenziamento dell'intelligenza piuttosto che di automazione. Oggi, Intelligenza Artificiale e Intelligenza Aumentata sono considerate due aspetti complementari, con la prima che enfatizza il comportamento intelligente della macchina e la seconda che sottolinea il controllo umano, entrambe essenziali per rendere le macchine utili agli esseri umani.

1.2.6 Ingegneria informatica

Come possiamo costruire computer più efficienti?
Per costruire un computer efficiente è fondamentale comprendere la sua evoluzione storica e tecnologica. Durante la Seconda Guerra Mondiale, i progressi nel campo della tecnologia informatica furono significativi, con l'invenzione indipendente del computer moderno in tre paesi.

- Il primo computer operativo fu l'Heath Robinson, costruito nel 1943 da Alan Turing e il suo team per decifrare i messaggi tedeschi.
- In seguito, il Colossus, sviluppato dallo stesso gruppo, fu il primo computer programmabile. Konrad Zuse in Germania creò il primo computer elettronico, lo Z-3, nel 1941, introducendo anche i numeri in virgola mobile e il linguaggio di programmazione Plankalkul.
- Il computer elettronico ABC, costruito da John Atanasoff e Clifford Berry tra il 1940 e il 1942, fu il precursore dell'ENIAC, sviluppato come parte di un progetto militare segreto.

Ogni generazione successiva di hardware informatico ha portato ad un aumento di velocità e capacità e ad una diminuzione dei costi, seguendo la legge di Moore[5]. Le aspettative future si concentrano sulla parallelizzazione massiccia e sull'ottimizzazione dell'hardware per applicazioni di Intelligenza Artificiale. L'avvento dell'informatica quantistica promette accelerazioni ancora maggiori per alcune importanti sottoclassi di algoritmi di Intelligenza Artificiale. Inoltre, il contributo di pionieri come Joseph Marie Jacquard e Charles Babbage ha posto le basi per la programmabilità e l'automazione, contribuendo al progresso dell'informatica.

1.2.7 Teoria del controllo e Cibernetica

Come possono gli artefatti operare sotto il proprio controllo?
Gli oggetti creati dall'uomo possono operare autonomamente grazie alla teoria dei sistemi

[5]La legge di Moore è un'osservazione empirica che afferma che il numero di transistor all'interno di un circuito integrato, e quindi la potenza di calcolo dei computer, raddoppia approssimativamente ogni due anni. Importante notare che negli ultimi anni, a causa di limitazioni tecnologiche e di progettazione, l'incremento della potenza di calcolo ha rallentato rispetto alla previsione originale della legge di Moore.

di controllo, che ha radici antiche. Un esempio avanguardistico è l'orologio ad acqua autoregolante di Ktesibios di Alessandria, che ha rivoluzionato la percezione delle capacità degli artefatti, permettendo loro di adattarsi all'ambiente come solo gli esseri viventi potevano fare in passato. Successivamente, il governatore del motore a vapore di James Watt e il termostato di Cornelis Drebbel hanno ampliato questa concezione. Norbert Wiener ha collegato la teoria dei sistemi di controllo ai sistemi biologici e meccanici, considerando il comportamento intenzionale come risultato di un meccanismo di regolazione che minimizza l'errore tra lo stato attuale e quello desiderato. Sebbene la moderna teoria dei sistemi di controllo miri a progettare sistemi che minimizzino una funzione di costo nel tempo, l'Intelligenza Artificiale (IA) e la teoria dei sistemi di controllo sono discipline separate, ciascuna con approcci matematici e problemi specifici, come la linguistica, la visione e la pianificazione simbolica, che vanno al di là delle competenze dei sistemi di controllo.

1.2.8 Linguistica

Come si relaziona il linguaggio al pensiero?
La linguistica moderna e l'Intelligenza Artificiale (IA) nacquero praticamente contemporaneamente e crebbero insieme, intersecandosi in un campo ibrido chiamato linguistica computazionale o elaborazione del linguaggio naturale. Si scoprì che il problema della comprensione del linguaggio era molto più complesso di quanto sembrasse nel 1957. Comprendere il linguaggio richiede una comprensione della materia e del contesto, non solo una comprensione della struttura delle frasi. Questo potrebbe sembrare ovvio, ma non fu ampiamente apprezzato fino agli anni '60. Molto del lavoro iniziale sulla rappresentazione della conoscenza (lo studio di come mettere la conoscenza in una forma con cui un computer possa ragionare) era legato al linguaggio e informato dalla ricerca in linguistica che, a sua volta, era collegata a decenni di lavoro sull'analisi filosofica del linguaggio.

1.3 Storia dell'Intelligenza Artificiale

La storia dell'Intelligenza Artificiale si può riassumere nella Figura 1.4.

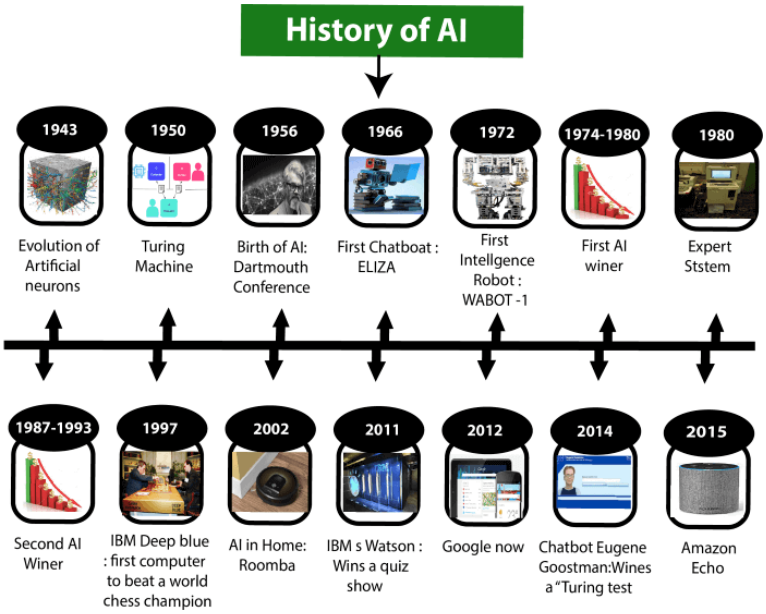


Figura 1.4: Immagine degli eventi più significativi della storia dell'IA

1.3.1 L'inizio dell'Intelligenza Artificiale (1943-1956)

Il primo lavoro generalmente riconosciuto come IA è stato realizzato da Warren McCulloch e Walter Pitts nel 1943. Essi si sono ispirati al lavoro di modellazione matematica del consigliere di Pitts, Nicolas Rashevsky, e hanno combinato conoscenze sulla fisiologia e il funzionamento dei neuroni cerebrali, l'analisi formale della logica proposizionale di Russell e Whitehead e la teoria della computazione di Turing. Hanno proposto un modello di neuroni artificiali in cui ogni neurone è caratterizzato come "acceso" o "spento", con uno switch "acceso" che si attiva in risposta a una stimolazione da parte di un numero sufficiente di neuroni vicini. McCulloch e Pitts hanno suggerito anche che reti opportunamente definite potessero imparare. Donald Hebb, nel 1949, ha dimostrato una semplice regola di aggiornamento per modificare le forze di connessione tra i neuroni, ora chiamata apprendimento hebbiano.

Marvin Minsky e Dean Edmonds, studenti universitari ad Harvard, hanno costruito nel 1950 il primo computer a rete neurale, chiamato SNARC. Altri esempi di lavori pionieristici includono due programmi per giocare a dama sviluppati indipendentemente nel 1952 da Christopher Strachey all'Università di Manchester e da Arthur Samuel all'IBM. Nel 1955, John McCarthy del Dartmouth College ha organizzato un workshop a Dartmouth, che ha riunito ricercatori interessati alla teoria degli automi, alle reti neurali e allo studio dell'intelligenza. Anche se non ha portato a scoperte significative, il workshop ha riunito alcuni dei precursori dell'IA e ha avviato la riflessione su come rendere le macchine capaci di simulare aspetti dell'apprendimento e dell'intelligenza umana.

1.3.2 Entusiasmo iniziale, grandi aspettative (1952-1969)

Negli anni '50, l'establishment intellettuale riteneva che le macchine non potessero compiere azioni specifiche. Gli studiosi di Intelligenza Artificiale (IA) risposero dimostrando abilità umane attraverso programmi informatici, come giochi, puzzle e test di intelligenza. Newell e Simon svilupparono il General Problem Solver (GPS), che imitava i protocolli umani di risoluzione dei problemi. Questo periodo fu chiamato l'era del "Guarda, mamma, senza mani!". Nel 1958, John McCarthy introdusse il linguaggio Lisp e propose l'Advice Taker, un programma ipotetico per il ragionamento basato sulla conoscenza. Successivamente, Marvin Minsky si unì al MIT dando inizio a ricerche sulla logica e sulle reti neurali. Nel frattempo, il lavoro di Arthur Samuel sul gioco degli scacchi dimostrò che i computer potevano imparare e migliorare autonomamente. Questo periodo di ricerca ebbe un impatto significativo sullo sviluppo futuro dell'IA, influenzando diverse aree di studio, dalle reti neurali alla logica e al ragionamento.

1.3.3 Una dose di realtà (1966-1973)

Fin dall'inizio, i ricercatori di Intelligenza Artificiale non si sono risparmiati nel fare previsioni sui loro imminenti successi. La seguente dichiarazione di Herbert Simon del 1957 viene spesso citata:

> *Non è mia intenzione sorprendervi o scioccarvi, ma il modo più semplice per riassumere è dire che esistono ora nel mondo macchine che pensano, imparano e creano. Inoltre, la loro capacità di fare queste cose aumenterà rapidamente fino a quando, in un futuro visibile, la gamma di problemi che possono gestire sarà coestensiva con la gamma a cui è stata applicata la mente umana.*

Il termine "futuro visibile" è vago, ma Simon fece anche previsioni più concrete: entro 10 anni un computer sarebbe stato campione di scacchi e un teorema matematico significativo sarebbe stato dimostrato da una macchina. Tuttavia, molte di queste previsioni non si sono

avverate nei tempi previsti, spesso a causa della sottostima della complessità dei problemi affrontati e della mancanza di considerazione delle limitazioni fondamentali delle strutture utilizzate per generare comportamenti intelligenti.

In definitiva, molte delle previsioni degli anni '50 e '60 sull'IA si sono rivelate troppo ottimistiche, ma hanno comunque gettato le basi per lo sviluppo futuro e l'avanzamento della ricerca nell'ambito dell'Intelligenza Artificiale.

1.3.4 Sistemi esperti (1969-1986)

Durante il periodo iniziale della ricerca sull'IA, si sono sviluppati due approcci principali: i "metodi deboli", che si basavano su una ricerca generale di soluzioni e non erano adatti a problemi complessi, e i sistemi basati sulla conoscenza del dominio, più efficaci perché permettevano passaggi di ragionamento più ampi e gestivano meglio i casi tipici.

Il programma DENDRAL è stato un esempio di successo dell'approccio basato sulla conoscenza del dominio, utilizzato per inferire la struttura molecolare da dati di spettrometria di massa. Questo ha aperto la strada a sistemi più avanzati, come MYCIN, utilizzato per diagnosticare infezioni del sangue.

Il successo di questi sistemi ha portato a una crescita delle applicazioni reali e allo sviluppo di strumenti di rappresentazione e ragionamento. Tuttavia, progetti su larga scala, come il "Quinta Generazione" giapponese e il progetto MCC negli Stati Uniti, non hanno raggiunto gli obiettivi prefissati.

Successivamente, molte aziende dell'IA hanno deluso nel fornire soluzioni a causa delle difficoltà nel gestire la complessità dei domini e dell'apprendimento dall'esperienza. Questo ha segnato l'inizio di un periodo noto come "inverno dell'IA".

1.3.5 Ragionamento probabilistico e "Machine learning" (1987-oggi)

La fragilità dei sistemi esperti ha dato vita a un nuovo approccio più scientifico, che predilige la probabilità alla logica booleana e l'apprendimento automatico alla codifica manuale. Questa nuova metodologia si basa su risultati sperimentali, anziché su affermazioni filosofiche, preferisce costruire su teorie esistenti, piuttosto che proporre nuove, e dimostrare la rilevanza per le applicazioni reali, anziché su esempi giocattolo.

Il 1988 ha segnato una svolta significativa, con l'adozione diffusa della probabilità e della teoria delle decisioni nell'IA grazie al lavoro di Judea Pearl. Inoltre, il collegamento dell'apprendimento per rinforzo alla teoria dei processi decisionali ha ampliato le applicazioni dell'IA in settori come la robotica e il controllo dei processi.

Questo nuovo approccio ha portato a una maggiore integrazione tra i vari sotto-campi dell'IA, migliorando sia le applicazioni pratiche che la comprensione teorica dei problemi fondamentali dell'IA.

1.3.6 Big data e Deep learning (2001-oggi)

Gli avanzamenti nella potenza di calcolo e l'espansione del World Wide Web hanno reso possibile la creazione di grandi set di dati, noti come big data, che includono testo, immagini, audio, video e altri dati. Ciò ha portato allo sviluppo di algoritmi di apprendimento in grado di sfruttare questi dati per ottenere risultati precisi, anche senza etichette su molti esempi. Questa tendenza ha dimostrato che l'incremento delle dimensioni del set di dati può portare a miglioramenti significativi nelle prestazioni degli algoritmi, superando le ottimizzazioni degli algoritmi stessi.

Inoltre, la visione artificiale ha beneficiato dell'ampia disponibilità di dati, come dimostrato dal miglioramento dei metodi di riempimento delle immagini attraverso l'uso di milioni

di immagini nel database ImageNet. L'uso dei big data ha riacceso l'interesse commerciale nell'Intelligenza Artificiale (IA), evidenziato dalla vittoria del sistema Watson di IBM nel quiz Jeopardy! del 2011.

Il concetto di deep learning, che impiega reti neurali artificiali con molti strati di neuroni, è emerso come una tecnologia rivoluzionaria nel riconoscimento vocale, visivo e in altri settori. L'impiego di hardware potente è cruciale per il successo del deep learning, che richiede, anche, grandi quantità di dati di addestramento. Questi successi hanno generato un rinnovato interesse nell'IA da parte di studenti, aziende, investitori e pubblico in generale, con una crescente attenzione sui progressi futuri e le potenziali implicazioni di questa tecnologia.

1.4 Tipi di Intelligenza Artificiale

Ci soffermiamo ora sui dettagli tecnici di questa disciplina. Nel vasto panorama dell'Intelligenza Artificiale, si delineano diverse tipologie, ognuna caratterizzata da specifiche capacità e funzionalità (Figura 1.5). Questa diversificazione consente di comprendere l'ampia gamma di applicazioni e approcci che contraddistinguono l'IA moderna. Analizzando le varie categorie, è possibile apprezzare la complessità e la ricchezza di questo campo, dove algoritmi, modelli e sistemi interconnessi delineano le sfumature dell'Intelligenza Artificiale nell'era digitale.

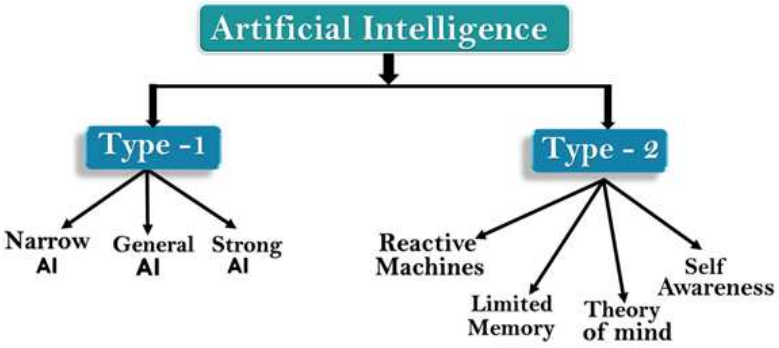


Figura 1.5: Immagine che riassume i vari tipi di IA

1.4.1 IA di tipo-1: Basata sulle capacità

- *IA Ristretta (o IA debole):* l'Intelligenza Artificiale ristretta, o Narrow AI, è un tipo di IA che si specializza nell'esecuzione di compiti specifici con un certo grado di intelligenza. È attualmente la forma più diffusa di IA nel panorama dell'Intelligenza Artificiale. Tuttavia, è limitata al compito per cui è stata progettata e addestrata, e non può operare al di là di questi limiti. Per questo motivo, viene spesso definita anche "debole". Un esempio comune di Narrow AI è Apple Siri, che funziona all'interno di un insieme predefinito di funzioni. Tuttavia, se l'IA ristretta viene sottoposta a compiti al di fuori del suo ambito, può comportarsi in modo imprevedibile.

  Alcune delle tipologie di IA debole che possiamo individuare sono:

  – *IA conversazionale:* progettata per interagire con gli utenti in modo simile a una conversazione umana; spesso utilizzata in chatbot, assistenti virtuali e sistemi di supporto clienti.

  - *IA predittiva:* utilizza algoritmi per analizzare dati passati e attuali al fine di fare previsioni o suggerire risultati futuri. Viene spesso impiegata nell'analisi dei dati, nel marketing predittivo e nella gestione delle risorse.
  - *IA generativa:* è in grado di creare nuovi contenuti, come immagini, testi, suoni o video, imitando lo stile e il contenuto dei dati di addestramento. È utilizzata in campo artistico, creativo e nella produzione di contenuti mediatici.
  - *IA autonoma:* è in grado di prendere decisioni e svolgere compiti senza l'intervento umano. È spesso utilizzata in applicazioni di robotica, veicoli autonomi, sistemi di gestione industriale e processi decisionali automatizzati.

- *IA Generale (o IA Forte):* l'Intelligenza Artificiale Generale è un tipo di intelligenza in grado di svolgere qualsiasi compito intellettuale con efficienza, proprio come farebbe un essere umano. L'obiettivo di fondo dell'IA generale è creare un sistema in grado di essere più intelligente e di pensare come un essere umano, in modo autonomo. Attualmente, non esiste ancora un sistema del genere che possa rientrare nell'ambito dell'IA generale e svolgere qualsiasi compito in modo perfetto come farebbe un essere umano. I ricercatori di tutto il mondo sono ora concentrati nello sviluppare macchine con Intelligenza Artificiale Generale. Poiché i sistemi con IA generale sono ancora in fase di ricerca, sarà necessario dedicare molto tempo ed energie per sviluppare tali sistemi.

- *Super IA:* la Superintelligenza Artificiale rappresenta un livello di intelligenza dei sistemi in cui le macchine potrebbero superare l'intelligenza umana ed eseguire qualsiasi compito con proprietà cognitive migliori degli esseri umani. Si tratta di un risultato dell'IA generale. Alcune caratteristiche chiave della superintelligenza artificiale includono la capacità di pensare, ragionare, risolvere enigmi, giudicare, pianificare, imparare e comunicare autonomamente. Tuttavia, la Superintelligenza Artificiale è ancora un concetto ipotetico nell'ambito dell'Intelligenza Artificiale. Lo sviluppo di tali sistemi nella realtà rimane un compito che potrebbe cambiare il mondo.

Nella Figura 1.6 riportiamo una immagine che mostra lo stato attuale dell'avanzamento nell'Intelligenza Artificiale.

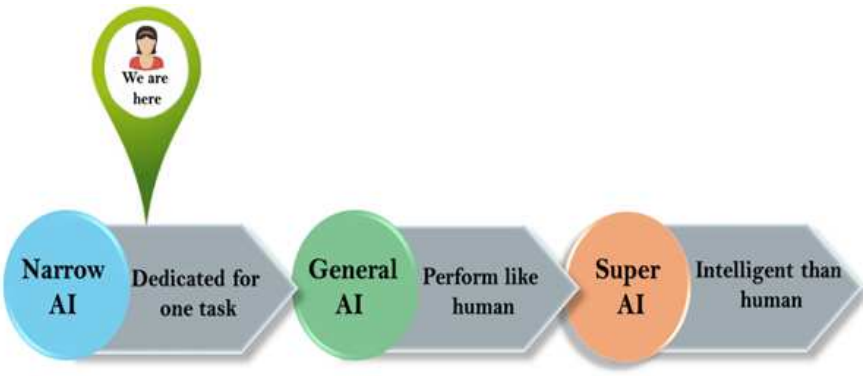


Figura 1.6: Immagine che mostra lo stato attuale dell'avanzamento dell'IA

1.4.2 IA di tipo-2: Basata sulle funzionalità

- *Macchine Reattive:* le macchine reattive, il livello più basilare di Intelligenza Artificiale, si focalizzano esclusivamente sugli scenari attuali, reagendo con l'azione considerata

ottimale. Non memorizzano ricordi o esperienze passate per decisioni future. Esempi includono il sistema Deep Blue di IBM e AlphaGo di Google.

- *Memoria Limitata:* queste macchine possono memorizzare esperienze passate o dati per un breve periodo, utilizzando le informazioni solo per una finestra temporale limitata. Le auto a guida autonoma sono un esempio, memorizzando informazioni come la velocità delle auto vicine e il limite di velocità per la navigazione.
- *Teoria della Mente:* questo tipo di IA dovrebbe comprendere emozioni umane, credenze e interagire socialmente come gli esseri umani. Sebbene ancora non esistano, i ricercatori stanno lavorando per svilupparle.
- *Auto consapevolezza:* l'IA autoconsapevole è un concetto ipotetico che rappresenta il futuro dell'Intelligenza Artificiale. Queste macchine, dotate di coscienza e autoconsapevolezza, supereranno la mente umana. Attualmente non esistono in forma reale, ma rappresentano un obiettivo futuro.

1.5 Modelli di apprendimento

I modelli di apprendimento dell'IA costituiscono un insieme di metodologie computazionali impiegate per istruire i sistemi intelligenti nell'identificare schemi, dedurre conclusioni e intraprendere azioni basate sui dati a loro disposizione. Essenziali per lo sviluppo dell'Intelligenza Artificiale, tali modelli comprendono una vasta gamma di tecniche e metodologie che fungono da fondamenta per la creazione e il funzionamento dei sistemi intelligenti (Figura 1.7).

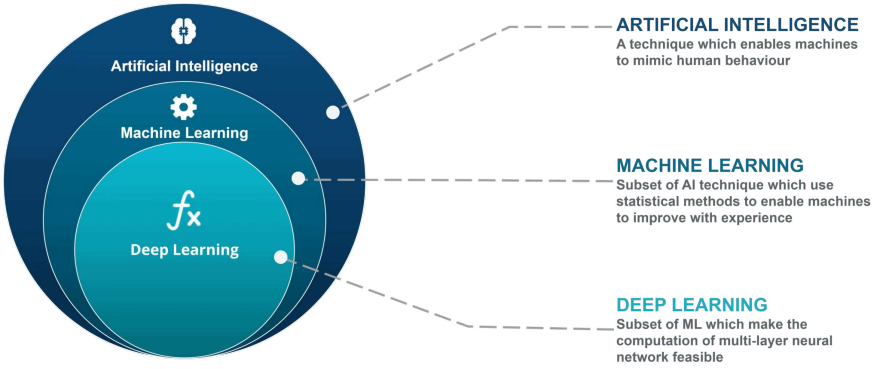


Figura 1.7: Immagine che fornisce una panoramica sui modelli di apprendimento

1.5.1 Machine Learning

Il machine learning rappresenta la disciplina scientifica che si occupa dello sviluppo di algoritmi e modelli statistici impiegati dai sistemi informatici per eseguire compiti senza ricevere istruzioni esplicite, basandosi, invece, su modelli e inferenze. In pratica, i sistemi informatici utilizzano gli algoritmi di machine learning per analizzare vaste quantità di dati storici al fine di identificare pattern e tendenze. Ciò consente loro di effettuare previsioni e predizioni più precise a partire da un insieme di dati iniziali.

Come funziona il machine learning?

Il concetto fondamentale alla base del machine learning risiede nella presenza di una relazione matematica tra un insieme di dati di input e di output. L'algoritmo di machine

learning non possiede a priori questa relazione, ma è in grado di dedurla se ad esso viene fornito un numero adeguato di set di dati. Ciò implica che ogni algoritmo di machine learning si basa su una funzione matematica modificabile.
Per comprendere questo principio, consideriamo il seguente esempio:

> Addestriamo l'algoritmo fornendo ad esso le seguenti coppie di input/output *(i, o): (2, 10), (5, 19) e (9, 31).* Secondo il calcolo dell'algoritmo, la relazione tra input e output è $o = 3 * i + 4$. Successivamente, gli forniamo un input di 7 e chiediamo di predire l'output. L'algoritmo è in grado di stabilire automaticamente che l'output corrisponde a 25.

Nonostante questa spiegazione sia di base, il machine learning si fonda sul principio che tutti i punti dati complessi possono essere collegati matematicamente dai sistemi informatici, purché siano disponibili dati sufficienti e potenza di calcolo per elaborarli. Di conseguenza, l'accuratezza dell'output dipende direttamente dalla quantità e dalla qualità dei dati di input forniti.

Quali sono i tipi di algoritmi di machine learning?

- *Machine learning con apprendimento supervisionato:* i data scientist[6] alimentano gli algoritmi con dati di addestramento che sono etichettati[7] e definiti, al fine di valutare le correlazioni. Questi dati di esempio specificano sia l'input che l'output dell'algoritmo. I vantaggi dell'apprendimento supervisionato risiedono nella sua semplicità e facilità di progettazione. È particolarmente utile per prevedere un numero limitato di risultati, suddividere i dati in categorie o integrare i risultati di altri due algoritmi di machine learning. Tuttavia, etichettare milioni di dati non etichettati rappresenta una sfida significativa.

- *Machine learning con apprendimento non supervisionato:* gli algoritmi di apprendimento non supervisionato si basano su dati non etichettati per individuare relazioni e modelli significativi tra input e output. Questo tipo di apprendimento è utile per riconoscere modelli, individuare anomalie e raggruppare automaticamente dati in categorie. Poiché non richiede dati etichettati, la configurazione è relativamente semplice. Tuttavia, non è in grado di fornire previsioni precise e non può isolare tipologie specifiche di risultati dei dati.

- *Machine learning con apprendimento semi-supervisionato*: questo metodo, noto come apprendimento semi-supervisionato, combina l'apprendimento supervisionato e non supervisionato. Si basa sull'utilizzo di un piccolo set di dati etichettati e un ampio set di dati non etichettati per addestrare il sistema. Inizialmente, i dati etichettati addestrano parzialmente l'algoritmo del machine learning, che successivamente etichetta autonomamente i dati non etichettati tramite il processo di pseudo-etichettatura. Il modello viene, poi, riaddestrato utilizzando questi nuovi dati, senza richiedere una programmazione esplicita.

[6] Un data scientist è un professionista specializzato nell'analisi dei dati che utilizza competenze statistiche, informatiche e di settore per estrarre informazioni significative e creare valore dalle grandi quantità di dati disponibili.

[7] L'etichettatura dei dati è il processo che associa i dati di input ai corrispondenti valori di output. Nei casi di apprendimento supervisionato, i dati di addestramento etichettati sono essenziali. Tuttavia, etichettare grandi quantità di dati può richiedere molto tempo e sforzo.

- *Machine learning con apprendimento per rinforzo:* l'apprendimento per rinforzo implica l'assegnazione di valori di ricompensa[8] a ciascun passo intrapreso dall'algoritmo, con l'obiettivo di massimizzare l'accumulo di punti di ricompensa e raggiungere un obiettivo finale. Questo approccio, ampiamente applicato nel settore dei videogiochi negli ultimi dieci anni, ha ottenuto risultati notevoli, superando talvolta le abilità umane in giochi classici e moderni. Sebbene sia efficace in ambienti con dati incerti e complessi, l'apprendimento per rinforzo è raramente implementato in contesti aziendali. Ciò è dovuto alla sua inefficienza nell'esecuzione di compiti specifici e alla suscettibilità all'influenza degli sviluppatori sulla progettazione delle ricompense, che può impattare significativamente sui risultati.

1.5.2 Deep Learning

Il deep learning è un sottoinsieme del machine learning; esso rappresenta un approccio di Intelligenza Artificiale (IA) che istruisce i computer a elaborare dati in modo analogo al cervello umano. I modelli di deep learning sono in grado di riconoscere schemi complessi in immagini, testo, suoni e altri tipi di dati per generare informazioni e previsioni precise. Grazie al deep learning, è possibile automatizzare attività che, solitamente, richiedono l'intervento umano, come la descrizione di immagini o la trascrizione di file audio in testo.

Come funziona il deep learning?

Gli algoritmi di deep learning sono basati su reti neurali progettate per emulare il funzionamento del cervello umano. Quest'ultimo, con milioni di neuroni interconnessi, lavora cooperativamente per acquisire e elaborare le informazioni. Analogamente, le reti neurali artificiali, costituenti del deep learning, sono composte da strati di neuroni artificiali che collaborano all'interno di sistemi informatici. I neuroni artificiali, chiamati nodi, sono componenti software che eseguono operazioni matematiche per elaborare i dati. Le reti neurali artificiali, (Figura 1.8), sono algoritmi di deep learning che sfruttano questi nodi per affrontare problemi complessi attraverso l'analisi dei dati.

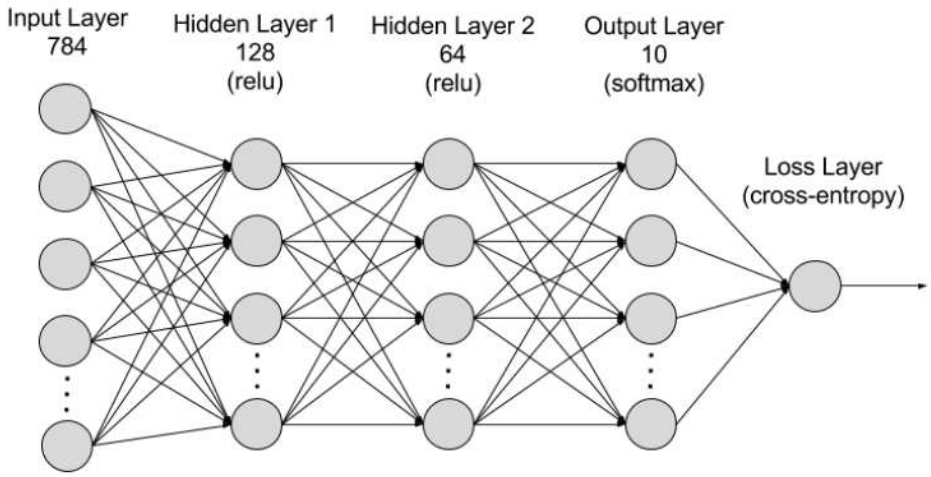


Figura 1.8: Immagine di una rete neurale

[8]Nel contesto dell'apprendimento per rinforzo, i "valori di ricompensa" si riferiscono agli incentivi o ai punteggi assegnati a ciascuna azione intrapresa dall'algoritmo durante il processo di apprendimento. Tali ricompense sono utilizzate dall'algoritmo per valutare la bontà delle azioni e guidare il processo di apprendimento.

Ora analizziamo quali sono i componenti di una rete neurale:

- *Livello di input:* una rete neurale artificiale ha diversi nodi che vi immettono dati. Questi ultimi costituiscono il livello di input del sistema.
- *Livello nascosto:* il layer di input elabora e trasmette i dati ai livelli successivi nella rete neurale. Questi livelli nascosti elaborano le informazioni a diversi livelli, adattando il loro comportamento man mano che ricevono nuove informazioni. Le reti di deep learning hanno centinaia di livelli nascosti che possono utilizzare per analizzare un problema da diverse angolazioni.
- *Livello di output:* il livello di output è costituito dai nodi che generano i dati. I modelli di deep learning che generano risposte "sì" o "no" hanno solo due nodi nel livello di output. D'altra parte, quelli che producono una gamma più ampia di risposte hanno più nodi.

1.5.3 Similitudini tra machine learning e deep learning

Tecniche di Intelligenza Artificiale

Il machine learning e il deep learning sono entrambi parte del campo più ampio della data science e dell'Intelligenza Artificiale (IA). Entrambi sono capaci di eseguire task computazionali complessi che, altrimenti, richiederebbero notevoli risorse e tempo con le tecniche di programmazione tradizionali.

Base statistica

Il deep learning e il machine learning utilizzano entrambi metodi statistici per addestrare i propri algoritmi con set di dati. Queste tecniche coinvolgono l'analisi della regressione, gli alberi decisionali, l'algebra lineare e il calcolo. Gli esperti di machine learning e gli esperti di deep learning conoscono bene la statistica.

Set di dati di grandi dimensioni

Tanto il machine learning quanto il deep learning richiedono insiemi considerevoli di dati di formazione di alta qualità per ottenere previsioni precise. Per esempio, un modello di machine learning tipicamente richiede circa 50-100 punti dati[9] per ciascuna caratteristica che cerca di apprendere. Al contrario, un modello di deep learning necessita di un numero molto maggiore di punti dati per caratteristica, spesso partendo da migliaia di punti dati.

Ampia gamma di applicazioni

Le soluzioni di machine learning e deep learning risolvono problemi complessi in tutti i settori e le applicazioni. Questi tipi di problemi richiederebbero molto più tempo per essere risolti o ottimizzati se si utilizzassero metodi di programmazione e statistici tradizionali.

[9]Questi "punti dati" rappresentano le osservazioni o gli esempi nel set di dati che vengono utilizzati per addestrare i modelli di machine learning o deep learning. Sono composti da una combinazione di variabili o caratteristiche che il modello cerca di comprendere e dalle rispettive etichette, o risultati associati, che il modello cerca di prevedere o classificare. La qualità e la quantità dei punti dati sono fondamentali per la precisione e l'efficacia dei modelli di apprendimento automatico e profondo.

Requisiti di potenza computazionale

Addestrare ed eseguire algoritmi di machine learning richiede una notevole potenza di calcolo e i requisiti computazionali sono ancora più elevati per il deep learning a causa della sua maggiore complessità. La disponibilità di entrambi per uso personale è ora possibile grazie ai recenti progressi nella potenza di calcolo e nelle risorse cloud.

Miglioramento graduale

Man mano che le soluzioni di machine learning e deep learning acquisiscono più dati, diventano più accurate nel riconoscimento dei pattern. Quando viene aggiunto un input al sistema, quest'ultimo lo utilizza come punto dati per l'addestramento, il che contribuisce al suo miglioramento.

1.5.4 Differenze tra machine learning e deep learning

Come riportato nella Sottosezione 1.5.2 il deep learning rappresenta un ramo del machine learning (ML) che si distingue per la sua complessità e potenza. Si tratta di una forma avanzata di machine learning che offre una vasta gamma di applicazioni. Tuttavia, i modelli di deep learning richiedono risorse più ampie, inclusi set di dati più estesi, infrastrutture più robuste e costi più elevati rispetto al machine learning tradizionale. Inoltre, mentre il machine learning spesso richiede un'interazione umana significativa per la progettazione e l'ottimizzazione dei modelli, il deep learning tende a eliminare questo bisogno grazie alla sua capacità di apprendere da grandi quantità di dati in modo autonomo. Questa necessità di evoluzione verso tecniche più avanzate come il deep learning è spinta dalla crescente complessità delle sfide e delle opportunità nel campo dell'Intelligenza Artificiale.

Ecco ulteriori differenze tra machine learning e deep learning.

Casi d'uso previsti

Il machine learning si presta perfettamente a compiti definiti con chiarezza, specialmente quando si tratta di dati organizzati e categorizzati.

D'altra parte, il deep learning si dimostra particolarmente efficace in ambiti complessi, dove le macchine devono elaborare dati non strutturati per estrarre significato e informazioni rilevanti.

Approccio alla risoluzione dei problemi

Il machine learning risolve problemi mediante l'applicazione di principi statistici e matematici. Al contrario, il deep learning si distingue per la sua approfondita fusione tra concetti statistici e matematici e l'utilizzo di un'architettura avanzata basata su reti neurali.

Metodi di addestramento

Il machine learning ha quattro metodi di formazione principali: apprendimento supervisionato, apprendimento non supervisionato, apprendimento semi-supervisionato e apprendimento per rinforzo. Altri metodi di formazione includono l'apprendimento per trasferimento e l'apprendimento supervisionato autonomo.

Al contrario, gli algoritmi di deep learning utilizzano diversi tipi di metodi di formazione più complessi. Queste includono reti neurali convoluzionali, reti neurali ricorrenti, reti generative contraddittorie e autocodificatori.

Coinvolgimento umano

Entrambe le metodologie di machine learning e deep learning richiedono un coinvolgimento umano significativo. È necessario definire il problema, preparare i dati, selezionare e addestrare un modello, valutarlo, ottimizzarlo e, infine, implementare una soluzione. I modelli di machine learning tendono ad essere più interpretabili per gli esseri umani, poiché si basano su modelli matematici più semplici, come gli alberi decisionali.

Diversamente, i modelli di deep learning sono più complessi e richiedono un'analisi dettagliata, in quanto sono matematicamente più sofisticati. Tuttavia, l'apprendimento delle reti neurali riduce la necessità di etichettare manualmente i dati; e in esso si può ridurre ulteriormente il coinvolgimento umano utilizzando modelli e piattaforme preaddestrati.

Requisiti dell'infrastruttura e prestazioni

I modelli di deep learning, per la loro complessità e le elevate dimensioni dei set di dati richiesti, necessitano di maggiori risorse di archiviazione e potenza di calcolo rispetto ai modelli di machine learning (ML). Mentre i dati e i modelli di ML possono essere gestiti su singole istanze o su cluster di server, il deep learning richiede spesso cluster ad alte prestazioni e altre infrastrutture robuste.

Gli investimenti in infrastruttura per le soluzioni di deep learning possono comportare costi significativamente superiori rispetto a quelli previsti dal machine learning. L'implementazione di infrastrutture locali potrebbe risultare poco pratica o poco conveniente per le soluzioni di deep learning. Tuttavia, è possibile mitigare tali costi utilizzando infrastrutture scalabili e servizi di deep learning completamente gestiti.

1.6 Rischi e benefici dell'Intelligenza Artificiale

L'ascesa dell'Intelligenza Artificiale (IA) in settori chiave come l'economia, la società, la scienza, la medicina, le finanze e la difesa richiede una ponderata valutazione dei rischi e dei benefici associati. Parlando in termini moderni, è fondamentale considerare gli aspetti positivi e negativi che l'IA può portare.

Esaminando i vantaggi, possiamo affermare che l'intera struttura della nostra civiltà è il frutto dell'intelligenza umana. Se avessimo accesso a un'intelligenza artificiale di livello superiore, le nostre prospettive si espanderebbero notevolmente. L'IA e la robotica hanno il potenziale di liberare l'umanità dalle mansioni ripetitive, incrementando la produzione di beni e servizi e aprendo le porte a un'epoca di prosperità e pace. Inoltre, la capacità di accelerare la ricerca scientifica potrebbe condurre a cure per le malattie e soluzioni per le sfide ambientali e le carenze di risorse, come suggerito da Demis Hassabis, CEO di Google DeepMind, che ha affermato: "Prima risolviamo l'IA, poi usiamo l'IA per risolvere tutto il resto". Tuttavia, molto prima di poter affrontare l'obiettivo di "risolvere l'IA", ci troviamo di fronte ai rischi derivanti dal suo uso improprio, sia intenzionale che accidentale. Alcuni di questi rischi sono già evidenti, mentre altri potrebbero emergere sulla base delle attuali tendenze:

- *Armi letali autonome:* queste sono definite dalle Nazioni Unite come armi in grado di individuare, selezionare ed eliminare bersagli umani senza intervento umano. Una preoccupazione primaria riguardo a tali armi è la loro scalabilità: l'assenza del requisito di supervisione umana significa che un piccolo gruppo può schierare un numero arbitrariamente grande di armi contro bersagli umani definiti da qualsiasi criterio di riconoscimento fattibile. Le tecnologie necessarie per le armi autonome sono simili a quelle necessarie per le auto a guida autonoma. Le discussioni informali degli esperti

sui potenziali rischi delle armi letali autonome sono iniziate presso le Nazioni Unite nel 2014, passando alla fase formale di pre-trattato di un Gruppo di Esperti Governativi nel 2017.

- *Sorveglianza e persuasione:* sebbene sia costoso, noioso, e a volte legalmente discutibile, per il personale di sicurezza monitorare linee telefoniche, feed di telecamere video, email e altri canali di messaggistica, l'IA (riconoscimento vocale, visione artificiale e comprensione del linguaggio naturale) può essere utilizzata in modo scalabile per eseguire una sorveglianza di massa degli individui e per individuare attività di interesse. Modificando i flussi di informazioni agli individui attraverso i social media, basandosi su tecniche di apprendimento automatico, il comportamento politico può essere modificato e controllato in qualche misura, una preoccupazione emersa nelle elezioni a partire dal 2016.

- *Decisioni tendenziose:* un uso negligente o intenzionale degli algoritmi di apprendimento automatico per compiti come la valutazione delle richieste di libertà vigilata e prestiti può portare a decisioni influenzate da razza, genere o altre categorie protette. Spesso, i dati stessi riflettono un pregiudizio diffuso nella società.

- *Impatto sull'occupazione:* le preoccupazioni riguardo alle macchine che eliminano posti di lavoro risalgono a secoli fa. La storia non è mai semplice: le macchine svolgono alcune delle attività che altrimenti potrebbero essere svolte dagli esseri umani, ma rendono anche gli esseri umani più produttivi, e quindi più impiegabili, e rendono le aziende più redditizie, e quindi in grado di pagare salari più alti. Possono rendere economicamente sostenibili alcune attività che altrimenti sarebbero impraticabili. Il loro uso generalmente porta a un aumento della ricchezza ma tende a spostare la ricchezza dal lavoro al capitale, aggravando ulteriormente l'aumento delle disuguaglianze. Avanzamenti tecnologici precedenti, come l'invenzione dei telai meccanici, hanno causato gravi interruzioni nell'occupazione, ma alla fine le persone trovano nuovi tipi di lavoro da fare. D'altra parte, è possibile che l'IA stia facendo anche quei nuovi tipi di lavoro. Questo argomento sta diventando rapidamente un focus principale per economisti e governi di tutto il mondo.

- *Applicazioni critiche per la sicurezza:* con l'avanzamento delle tecniche di Intelligenza Artificiale, queste vengono sempre più utilizzate in applicazioni critiche per la sicurezza, come la guida delle auto e la gestione delle risorse idriche delle città. Incidenti fatali sono già accaduti ed evidenziano la difficoltà della verifica formale e dell'analisi statistica del rischio per i sistemi sviluppati utilizzando tecniche di apprendimento automatico. Il campo dell'IA dovrà sviluppare standard tecnici ed etici almeno comparabili a quelli prevalenti in altre discipline ingegneristiche e sanitarie dove sono in gioco le vite delle persone;

- *Sicurezza informatica:* le tecniche di intelligenza artificiale sono utili nella difesa contro gli attacchi informatici, ad esempio rilevando schemi di comportamento insolito, ma contribuiranno anche alla potenza, alla sopravvivenza e alla capacità di proliferazione del malware[10]. Ad esempio, metodi di apprendimento per rinforzo sono stati utilizzati per creare strumenti altamente efficaci per attacchi di ricatto e phishing automatizzati e personalizzati.

[10] È un termine generico che indica un software progettato per danneggiare, compromettere o infiltrarsi in un sistema informatico senza il consenso dell'utente. Può includere virus, worm, trojan, spyware, adware e altri tipi di software dannoso.

Tutti questi rischi sottolineano l'importanza della governance e, alla fine, della regolamentazione dell'IA. Al momento, la comunità scientifica e le principali aziende coinvolte nella ricerca sull'IA hanno sviluppato principi di autoregolamentazione volontaria per le attività legate all'IA. Governi e organizzazioni internazionali stanno istituendo organi consultivi per elaborare regolamenti appropriati per ogni caso d'uso specifico e per prepararsi agli impatti economici e sociali dell'IA.

Nel lungo periodo, l'interrogativo fondamentale riguarda la possibilità di raggiungere l'ambizioso obiettivo di creare un'intelligenza artificiale che sia almeno paragonabile, se non superiore, a quella umana. Questa prospettiva, per lungo tempo trascurata nella storia dell'IA, sta emergendo sempre più come tema cruciale. La preoccupazione principale è legata alla creazione di una superintelligenza artificiale (ASI), capace di superare nettamente le capacità umane, suscitando, quindi, una riflessione profonda sulla gestione dell'IA e sui suoi potenziali rischi.

I timori riguardanti l'ASI sono stati sollevati da diverse fonti autorevoli, tra cui Yudkowsky (2008) e Omohundro (2008). Già nel lontano 1951, Turing ha posto l'accento su un concetto analogo durante una conferenza a Manchester: una volta avviato il processo di pensiero delle macchine, potrebbe richiedere poco tempo prima che esse superino le capacità limitate dell'umanità. Ciò apre la porta all'eventualità che le macchine assumano il controllo, tema che ha guadagnato ulteriore rilevanza con i recenti progressi nel deep learning e la pubblicazione di testi come "Superintelligence" di Nick Bostrom (2014), nonché le dichiarazioni di esperti come Stephen Hawking, Bill Gates, Martin Rees ed Elon Musk.

L'idea di creare macchine superintelligenti suscita un diffuso senso di disagio. Se la creazione di un'ASI comportasse la perdita di controllo dell'umanità sul proprio destino, si porrebbe una seria questione etica e pratica. Questo è il fulcro dell'avvertimento di Turing: il controllo su macchine più intelligenti di noi non è affatto scontato.

Se si considera l'ASI come un enigma proveniente dallo spazio, l'adozione di misure di cautela nell'approccio a questa tecnologia sarebbe senz'altro prudente. Tuttavia, essendo noi i progettisti di sistemi di IA, se mai dovessero "prendere il controllo", come suggerisce Turing, sarebbe il risultato di un errore di progettazione da parte nostra. Di conseguenza, è essenziale comprendere appieno le potenziali fonti di fallimento e agire di conseguenza per mitigare i rischi associati a un simile scenario.

CAPITOLO 2

Amazon AWS

In questo capitolo verrà presentato uno dei cloud più completi e utilizzati al mondo, ovvero AWS (Amazon Web Services). In particolare, si parlerà in dettaglio di che cos'è AWS, di come funziona e dei vari servizi che offre. Riguardo a quest'ultimo aspetto il nostro focus ricadrà sui servizi che sono stati utilizzati per affrontare i vari case study che saranno discussi nei capitoli successivi.

2.1 Un'introduzione al concetto di Cloud

I servizi cloud sono risorse infrastrutturali, piattaforme o software ospitati da provider esterni e resi accessibili agli utenti tramite Internet. Questi servizi consentono agli utenti di gestire ed elaborare dati, nonché di eseguire applicazioni senza dover possedere o gestire direttamente l'hardware o il software sottostante.

In altre parole, anziché dover installare e mantenere fisicamente server, sistemi operativi o software sul proprio hardware locale, gli utenti possono accedere a risorse informatiche remote attraverso il cloud. Ciò significa che i dati e le applicazioni possono essere memorizzati, elaborati e gestiti su server remoti, consentendo agli utenti di accedervi da qualsiasi luogo connesso a Internet.

I servizi cloud agevolano il flusso dei dati dagli utenti ai server remoti, e viceversa, consentendo una comunicazione efficiente e veloce tra i dispositivi degli utenti e i sistemi dei provider cloud. Questi servizi incentivano la creazione di applicazioni progettate specificamente per essere eseguite nel cloud, sfruttando le sue caratteristiche di scalabilità, affidabilità e disponibilità.

Per accedere ai servizi cloud, gli utenti hanno bisogno solo di un computer con un sistema operativo e una connessione Internet. Ciò permette una maggiore flessibilità nel lavoro e nell'elaborazione dei dati, poiché non sono necessarie risorse hardware locali e le risorse cloud possono essere facilmente scalate in base alle esigenze dell'utente.

2.1.1 Tipologie di servizi cloud

Le tipologie di servizi che vengono fornite da un cloud provider sono le seguenti:

- *Infrastructure as a Service (IaaS - Infrastruttura come Servizio):* i servizi IaaS forniscono agli utenti l'accesso a risorse di calcolo, come server virtuali, reti, storage e macchine virtuali, ospitati nel cloud. Gli utenti possono gestire e configurare queste risorse secondo le proprie esigenze, senza dover gestire l'hardware fisico sottostante.

- *Platform as a Service (PaaS - Piattaforma come Servizio):* PaaS fornisce agli sviluppatori un ambiente completo per sviluppare, testare e distribuire applicazioni software senza dover gestire l'infrastruttura sottostante. Questi servizi includono piattaforme di sviluppo, database, strumenti di analisi e framework di gestione delle applicazioni. PaaS semplifica il processo di sviluppo e distribuzione delle applicazioni, consentendo agli sviluppatori di concentrarsi sulla scrittura del codice senza preoccuparsi dell'infrastruttura.

- *Software as a Service (SaaS - Software come Servizio):* SaaS offre agli utenti accesso a software e applicazioni ospitati nel cloud, accessibili attraverso Internet tramite un browser web o un'interfaccia utente. Gli utenti non devono installare o gestire il software localmente, ma possono utilizzarlo su richiesta.

- *Function as a Service (FaaS - Funzione come Servizio):* FaaS è un modello di servizio cloud in cui gli sviluppatori possono scrivere e distribuire singole funzioni o pezzi di codice senza dover gestire l'infrastruttura sottostante. Le funzioni vengono eseguite su richiesta in risposta agli eventi, con il cloud che gestisce automaticamente il provisioning e la scalabilità delle risorse necessarie per eseguire il codice. Questo approccio è particolarmente utile per lo sviluppo di microservizi e applicazioni serverless.

2.2 Storia ed evoluzione di AWS

Amazon Web Services (AWS) è stata lanciata ufficialmente nel 2006 da Amazon.com Inc., inizialmente come una divisione interna dedicata alla fornitura di servizi di infrastruttura IT attraverso il cloud. Tuttavia, il processo che ha portato alla creazione di AWS ha radici molto più profonde nella storia di Amazon.

Tutto è iniziato quando Amazon si è trovata di fronte a un'enorme crescita del proprio business online e ha dovuto affrontare la sfida di gestire e scalare rapidamente le proprie infrastrutture IT per soddisfare la domanda dei clienti. Per far fronte a questa sfida, Amazon ha sviluppato internamente una serie di tecnologie e strumenti per gestire la propria infrastruttura in modo efficiente e scalabile.

Basandosi sull'esperienza acquisita durante la gestione della propria infrastruttura, Amazon ha deciso di mettere a disposizione del pubblico esterno una serie di servizi cloud, inizialmente focalizzati su risorse di calcolo e storage, come Amazon Elastic Compute Cloud (EC2) e Amazon Simple Storage Service (S3). Questi servizi hanno permesso agli sviluppatori di accedere a risorse di calcolo e di storage su richiesta, senza dover gestire l'hardware sottostante.

Negli anni successivi, AWS ha continuato a espandere il proprio portfolio di servizi, introducendo una vasta gamma di soluzioni per calcolo, archiviazione, database, analisi, Intelligenza Artificiale, machine learning, sicurezza, sviluppo di applicazioni, IoT (Internet delle cose), blockchain, e molto altro. AWS è diventata rapidamente leader nel settore del cloud computing, offrendo servizi affidabili, scalabili e altamente performanti, che hanno rivoluzionato il modo in cui le aziende costruiscono e gestiscono le proprie applicazioni e infrastrutture IT.

2.3 Architettura e infrastruttura di AWS

AWS offre un'infrastruttura globale estesa che è stata riconosciuta da Gartner[1] come l'approccio ideale per eseguire applicazioni aziendali che richiedono un'elevata disponibilità.

Questa infrastruttura è distribuita strategicamente attraverso varie regioni geografiche, ciascuna composta da almeno due o più zone di disponibilità. Le regioni AWS sono progettate per essere indipendenti l'una dall'altra, garantendo la continuità dei servizi anche in caso di interruzioni in una di esse.

All'interno di ciascuna regione, le zone di disponibilità rappresentano località fisiche isolate, dotate di infrastrutture autonome per garantire la robustezza e l'affidabilità dei servizi. Questo design consente agli utenti di distribuire le proprie applicazioni in modo resiliente su una base geograficamente distribuita.

Inoltre, AWS dispone di edge location distribuite in tutto il mondo, utilizzate per la distribuzione di contenuti e l'accelerazione dei servizi attraverso una Content Delivery Network (CDN). Le edge location sono punti strategicamente posizionati nella rete AWS ottimizzati per la distribuzione di contenuti a bassa latenza, garantendo che i dati raggiungano gli utenti rapidamente.

In sintesi, l'infrastruttura globale di AWS (Figura 2.1), composta da regioni, zone di disponibilità e edge location, offre ai clienti un'esperienza affidabile, scalabile e performante nel cloud computing, sostenendo le esigenze di una vasta gamma di applicazioni e carichi di lavoro.

L'infrastruttura centrale è progettata per soddisfare i requisiti di sicurezza per organizzazioni ad elevata sensibilità, come quelle militari, bancarie globali e molto altro. Ciò è supportato da un set completo di strumenti di sicurezza cloud, con oltre 300 servizi e funzionalità di sicurezza, conformità e governance, oltre al supporto per 143 standard di sicurezza e certificazioni di conformità.

Figura 2.1: La rete globale di AWS

[1] Gartner Inc. è una società per azioni multinazionale che si occupa di consulenza strategica, ricerca di mercato e analisi nel campo della tecnologia dell'informazione.

2.4 Servizi AWS

Per esplorare a fondo i vari servizi AWS che ci interessano, è utile introdurre alcune nozioni che ci permetteranno di comprendere meglio il loro funzionamento di base.

Natural Language Processing (NLP)

L'elaborazione del linguaggio naturale è un campo dell'Intelligenza Artificiale che si occupa di consentire ai computer di comprendere, interpretare e generare linguaggio umano in modo naturale. L'NLP si concentra sull'interazione tra il linguaggio umano e i sistemi informatici, consentendo ai computer di comprendere il significato del linguaggio umano attraverso una serie di tecniche e algoritmi.

Le principali attività dell'NLP includono:

1. *Tokenizzazione:* suddivisione del testo in parole, frasi o altre unità significative.
2. *Analisi della morfologia:* identificazione della forma e della struttura delle parole.
3. *Analisi sintattica:* analisi della struttura grammaticale delle frasi per comprendere le relazioni tra le parole.
4. *Analisi semantica:* interpretazione del significato delle parole e delle frasi all'interno di un contesto specifico.
5. *Generazione del linguaggio:* creazione di testo umano in modo coerente e comprensibile.

L'NLP è ampiamente utilizzato in una varietà di applicazioni, tra cui traduzione automatica, analisi dei sentimenti, estrazione di informazioni, assistenti virtuali e chatbot.

Natural Language Understanding (NLU)

La comprensione del linguaggio naturale è una branca dell'Intelligenza Artificiale e dell'elaborazione del linguaggio naturale (NLP) che si concentra sull'interpretazione e sulla comprensione del linguaggio umano da parte dei computer. L'NLU consente ai computer di comprendere e interpretare in modo significativo il linguaggio umano, permettendo loro di analizzare il significato di frasi, domande o testi.

L'NLU si occupa di diverse attività, tra cui:

1. *Estrazione di informazioni:* estrazione di informazioni rilevanti da testi, come entità, eventi, date o luoghi.
2. *Classificazione di intenti:* classificazione delle intenzioni o degli obiettivi dietro una frase o una domanda, consentendo ai sistemi di rispondere in modo appropriato.
3. *Analisi del sentimento:* determinazione del tono o del sentimento di un testo, per individuare se è positivo, negativo o neutro;
4. *Comprensione del contesto:* può comprendere il contesto di una conversazione o di un testo, aiutando i sistemi a fornire risposte più accurate e pertinenti.

L'NLU è una tecnologia fondamentale utilizzata in una vasta gamma di applicazioni, tra cui assistenti virtuali, chatbot, motori di ricerca, sistemi di analisi dei social media e traduttori automatici.

Automatic Speech Recognition (ASR)

Il riconoscimento vocale automatico è una tecnologia che consente ai computer di comprendere e interpretare il linguaggio umano parlato. L'ASR converte il parlato in input in testo scritto utilizzando algoritmi e modelli di machine learning.

Le principali fasi del processo di ASR includono:

1. *Acquisizione audio:* il suono viene registrato attraverso microfoni o altri dispositivi audio.
2. *Pre-elaborazione:* l'audio può essere elaborato per migliorare la qualità e ridurre il rumore di fondo.
3. *Segmentazione:* l'audio viene suddiviso in segmenti più piccoli, come frasi o singole parole.
4. *Caratterizzazione:* ogni segmento audio viene analizzato per estrarre le caratteristiche acustiche, come la frequenza e l'intensità del suono.
5. *Riconoscimento:* utilizzando algoritmi di riconoscimento del linguaggio, il suono viene confrontato con modelli linguistici per determinare la sequenza di parole più probabile.
6. *Decodifica:* l'output del riconoscimento viene decodificato in forma di testo.

L'ASR è impiegato diffusamente in una vasta gamma di contesti applicativi , come i sistemi di controllo vocale (quali Siri di Apple, Google Assistant, Amazon Alexa), la trascrizione automatica di audio e video e i sistemi di traduzione simultanea.

2.4.1 Amazon Lex

Amazon Lex è un servizio progettato per creare interfacce di comunicazione per le applicazioni utilizzando sia la voce che il testo. Basato sulla stessa tecnologia di deep learning di Amazon Alexa, Amazon Lex offre funzionalità avanzate e flessibilità nel riconoscimento del linguaggio naturale (NLU) e nel riconoscimento vocale automatico (ASR), permettendo di sviluppare applicazioni coinvolgenti con conversazioni realistiche e di aprire nuove possibilità in termini di prodotti e servizi.

Amazon Lex semplifica la creazione di bot conversazionali per sviluppatori di ogni livello. Non è necessaria alcuna esperienza di deep learning per creare un bot; basta specificare il flusso di conversazione di base. Il servizio si occupa automaticamente del dialogo e regola dinamicamente le risposte nella conversazione. Tramite la console, è possibile creare, testare e pubblicare chatbot di testo o vocali. Inoltre, è possibile integrare le interfacce di comunicazione dei bot su dispositivi mobili, applicazioni Web e piattaforme di chat, come Facebook e Messenger (Figura 2.2).

Concetti e terminologia di base di Amazon Lex

- *Lingua:* un bot Amazon Lex può conversare in una o più lingue; è possibile configurare Amazon Lex per conversare con un utente utilizzando parole e frasi native.
- *Intento:* è un'astrazione dei desideri o delle azioni che gli utenti possono esprimere attraverso le loro interazioni con il bot di Amazon Lex. Gli intenti rappresentano le intenzioni dell'utente di eseguire una specifica azione o di ottenere una determinata informazione. Ogni intento è associato a una serie di frasi che gli utenti potrebbero pronunciare per esprimere quell'intenzione. Quando l'utente interagisce con il chatbot,

Figura 2.2: Schema generale di Amazon Lex

Amazon Lex utilizza le tecniche di comprensione del linguaggio naturale per identificare l'intento dell'utente e rispondere di conseguenza, eseguendo le azioni appropriate o fornendo le informazioni richieste.
Per ogni intento, devono essere forniti:

- il nome;
- esempi di enunciati con cui l'utente potrebbe trasmette l'intento (Utterances);
- le modalità con cui deve essere soddisfatto l'intento (Fulfillment).

- *Slot:* rappresenta una variabile che può essere utilizzata per raccogliere e memorizzare informazioni specifiche fornite dall'utente durante l'interazione con il bot. Gli slot vengono utilizzati per estrarre e interpretare i valori dei parametri rilevanti da una richiesta dell'utente, consentendo al bot di comprendere meglio l'intento dell'utente e di fornire risposte o eseguire azioni appropriate. Ad esempio, durante una conversazione con un bot per la prenotazione di un volo, gli slot potrebbero essere utilizzati per raccogliere informazioni come la città di partenza, la destinazione, la data e l'orario del volo desiderato.

- *Tipi di slot*: in Amazon Lex, ci sono diverse tipologie di slot per soddisfare al meglio l'intento dell'utente. Ecco alcune delle principali tipologie di slot:

  - *Slot di testo:* gli slot di testo vengono utilizzati per raccogliere informazioni testuali fornite dall'utente, come nomi, indirizzi, numeri di telefono, e così via.
  - *Slot numerici:* gli slot numerici vengono utilizzati per raccogliere valori numerici, come quantità, età, o qualsiasi altra informazione numerica richiesta durante l'interazione.
  - *Slot di data/ora:* questi slot vengono utilizzati per raccogliere informazioni relative a date e orari, come date di prenotazione, scadenze, orari di appuntamenti, e così via.
  - *Slot di elenco:* gli slot di elenco consentono all'utente di selezionare una voce da un elenco predefinito di opzioni, ad esempio, un elenco di città, categorie di prodotti, oppure opzioni di pagamento.
  - *Slot di conferma:* gli slot di conferma sono utilizzati per raccogliere informazioni che richiedono conferma da parte dell'utente, come "Sì" o "No", per garantire la correttezza delle informazioni raccolte.

  - *Slot personalizzati:* Amazon Lex consente agli sviluppatori di definire slot personalizzati per gestire casi specifici o complessi che non rientrano nelle categorie standard di slot. Questi slot possono essere configurati per raccogliere informazioni secondo criteri personalizzati definiti dagli sviluppatori.

- *Versione:* una versione è un'istantanea numerata dello stato attuale del bot che è possibile pubblicare per utilizzarla in diverse parti del flusso di lavoro, come sviluppo, distribuzione beta e produzione.

- *Alias:* un alias indica una versione specifica di un bot; con un alias, è possibile aggiornare la versione utilizzata dalle applicazioni client.

  Alcuni dei concetti fondamentali per la creazione di un bot conversazionale sono riportati in Figura 2.3.

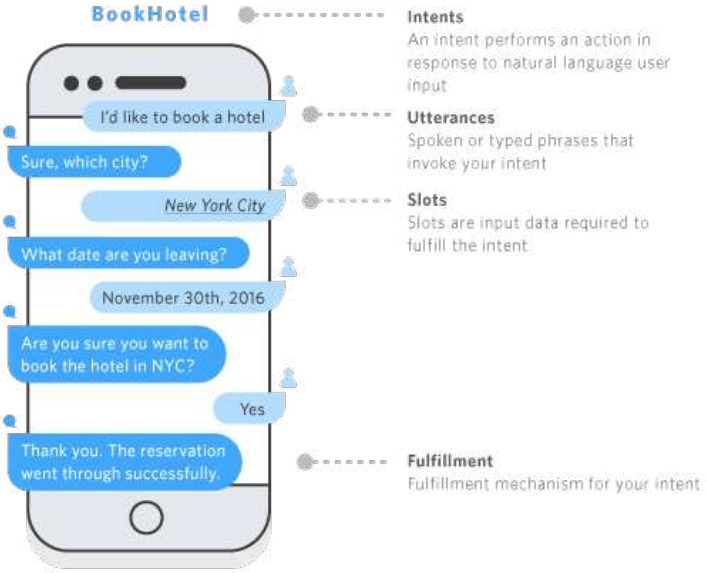


Figura 2.3: Concetti base di Amazon Lex

Vantaggi di Amazon Lex:

Amazon Lex presenta i seguenti vantaggi:

- *Semplicità:* il servizio semplifica il processo di creazione del nostro bot attraverso un'esperienza guidata nella console. Fornendo alcune frasi di esempio Amazon Lex genererà un modello completo di linguaggio naturale. Quest'ultimo consentirà al bot di interagire sia tramite voce che mediante testo, permettendo ad esso di porre domande, ottenere risposte e completare attività sofisticate in modo intuitivo.

- *Tecnologie di deep learning:* il riconoscimento vocale e la comprensione del linguaggio naturale rappresentano alcune delle sfide più complesse in informatica, richiedendo sofisticati algoritmi di deep learning addestrati su grandi quantità di dati e infrastrutture. Amazon Lex rende accessibili tali tecnologie avanzate a tutti gli sviluppatori. I bot di Amazon Lex trasformano la voce in ingresso in testo e interpretano l'intento dell'utente per generare risposte intelligenti; ciò permette di concentrarsi sulla creazione del bot offrendo un valore aggiunto.

- *Implementazione e scalabilità senza interruzioni:* con Amazon Lex, è possibile creare, testare e distribuire i nostri bot direttamente dalla console. Amazon Lex si ridimensiona automaticamente e si occupa del provisioning dell'hardware e della gestione dell'infrastruttura per migliorare l'esperienza con i bot.

- *Integrazione integrata con la piattaforma AWS:* Amazon Lex si integra in modo nativo con altri servizi AWS, consentendo di sfruttare appieno la potenza della piattaforma AWS per una serie di funzionalità cruciali. È possibile beneficiare della robusta infrastruttura di sicurezza, del monitoraggio dettagliato, dell'autenticazione degli utenti, della logica di business personalizzata, dello storage affidabile e dello sviluppo di applicazioni mobili.

2.4.2 Amazon Comprehend

Amazon Comprehend, un servizio di elaborazione del linguaggio naturale (NLP), si basa su un modello pre-addestrato per analizzare e comprendere documenti senza la necessità di dati di addestramento aggiuntivi. Utilizzando l'Intelligenza Artificiale, Amazon Comprehend esamina il testo in modo automatico e scalabile, identificando entità, frasi chiave, sentimenti e altri elementi comuni. In questo modo, fornisce un'analisi completa e approfondita del contenuto testuale, rendendo più accessibili e utilizzabili le informazioni estratte dai documenti o set di documenti analizzati (Figura 2.4).

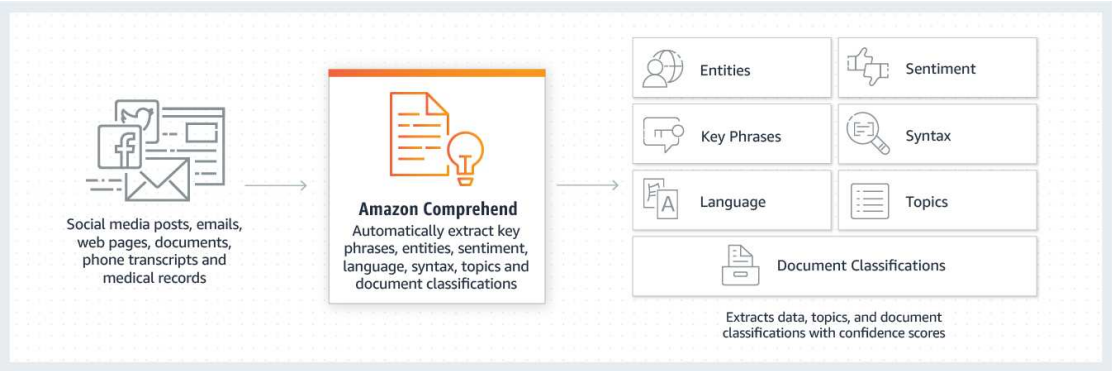


Figura 2.4: Schema generale di Amazon Comprehend

Concetti e funzionalità di base di Amazon Comprehend

Per ottenere una comprensione più approfondita del funzionamento dei concetti che esamineremo, prendiamo in considerazione un testo di riferimento e valutiamo le risposte fornite dalle diverse funzionalità di Comprehend.

> *Hello Zhang Wei, I am John. Your AnyCompany Financial Services, LLC credit card account 1111-0000-1111-0008 has a minimum payment of $24.53 that is due by July 31st. Based on your autopay settings, we will withdraw your payment on the due date from your bank account number XXXXXX1111 with the routing number XXXXX0000. Customer feedback for Sunshine Spa, 123 Main St, Anywhere. Send comments to Alice at sunspa@mail.com. I enjoyed visiting the spa. It was very comfortable but it was also very expensive. The amenities were ok but the service made the spa a great experience.*

- *Entità:* un'entità è un riferimento testuale al nome univoco di un oggetto del mondo reale, come persone, luoghi e articoli commerciali, e a riferimenti precisi a misure, come

date e quantità. Ogni entità ha, anche, un punteggio che indica il livello di fiducia di Amazon Comprehend rispetto al corretto rilevamento del tipo di entità. Nelle Figure 2.5 e 2.6 sono mostrati i risultati dell'analisi delle entità.

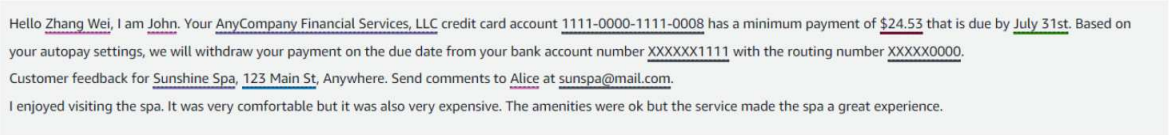


Figura 2.5: Rilevamento delle entità nel testo

Entity	Type	Confidence
Zhang Wei	Person	0.99+
John	Person	0.99+
AnyCompany Financial Services, LLC	Organization	0.99+
1111-0000-1111-0008	Other	0.99+
$24.53	Quantity	0.99+
July 31st	Date	0.99+
XXXXXX1111	Other	0.98
XXXXX0000	Other	0.97
Sunshine Spa	Organization	0.98
123 Main St	Location	0.98
Alice	Person	0.99+
sunspa@mail.com	Other	0.99+

Figura 2.6: Risultati dell'analisi delle entità

- *Frasi chiave:* sono frasi che rappresentano concetti o informazioni rilevanti e significative presenti nel documento. Esse sono identificate utilizzando algoritmi di analisi del linguaggio naturale che cercano le parole o le combinazioni di parole più importanti e indicative nel contesto del testo analizzato. Le "key phrases" forniscono un riassunto essenziale del contenuto del documento e possono essere utilizzate per estrarre informazioni cruciali, classificare i documenti o svolgere altre attività di analisi del testo (Figura 2.7).

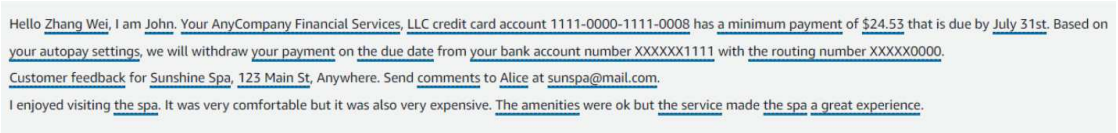


Figura 2.7: Risultati dell'analisi delle frasi chiave

- *Informazioni di identificazione personale (PII):* è un'opzione che consente di individuare e identificare informazioni personali sensibili all'interno di testi, come nomi, indirizzi, numeri di telefono, indirizzi email e numeri di carta di credito. Questo è utile per proteggere la privacy e garantire la conformità normativa, consentendo alle organizzazioni di identificare e proteggere le informazioni personali sensibili all'interno dei loro dati (Figura 2.8).
  Il suo funzionamento è riassunto nei seguenti task:

- *Riconoscimento dei dati sensibili:* Comprehend utilizza algoritmi di machine learning per analizzare il testo e individuare eventuali pattern che corrispondono a informazioni personali sensibili.
- *Classificazione dei dati sensibili:* una volta individuate, le informazioni personali sensibili vengono classificate in categorie specifiche, come nomi, indirizzi, numeri di telefono, indirizzi email, numeri di carta di credito, e così via.
- *Risultati dell'analisi:* Comprehend restituisce i risultati dell'analisi indicando quali parti del testo contengono informazioni personali identificabili e la tipologia di PII individuata.
- *Integrazione con altri servizi:* i risultati dell'analisi possono essere utilizzati per prendere decisioni informate sul trattamento dei dati sensibili, come la mascheratura o l'eliminazione di tali informazioni prima di archiviare o condividere i dati.

Entity	Type	Confidence
Zhang Wei	Name	0.99+
John	Name	0.99+
1111-0000-1111-0008	Credit debit number	0.99+
July 31st	Date time	0.99+
XXXXXX1111	Bank account number	0.99+
XXXXX0000	Bank routing	0.99+
123 Main St	Address	0.99+
Alice	Name	0.99+
sunspa@mail.com	Email	0.99+

Figura 2.8: Risultati dell'analisi dei PII

- *Lingua dominante:* Amazon Comprehend permette di determinare la lingua predominante in un documento o in un insieme di testi. Questa informazione è utile per comprendere il contesto e le caratteristiche linguistiche dei documenti e può essere impiegata per vari scopi, come l'organizzazione e la classificazione dei dati multilingue, l'analisi dei sentimenti, la traduzione automatica, e altro ancora (Figura 2.9).

Figura 2.9: Risultati dell'analisi della lingua dominante

- *Sentimento:* l'analisi del sentimento consente di determinare il tono emotivo di un testo. Questa funzione valuta se il testo è positivo, negativo o neutro, consentendo di comprendere meglio l'opinione o il sentimento associato al testo analizzato. Le operazioni di analisi restituiscono il sentimento più probabile per il testo e i punteggi per ciascuno dei sentimenti. Il punteggio rappresenta la probabilità che l'emozione sia stata rilevata correttamente (Figura 2.10).

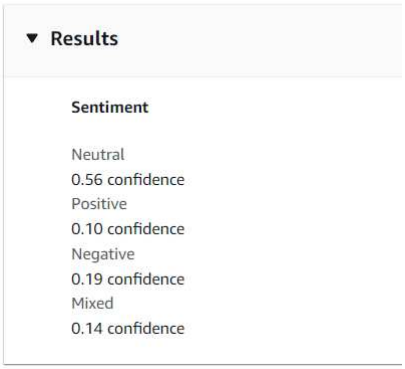


Figura 2.10: Risultati dell'analisi del sentimento

- *Sentimento mirato:* il sentimento mirato fornisce una comprensione dettagliata dei sentimenti associati a entità specifiche nei documenti di input. La differenza tra l'analisi del sentimento mirato e l'analisi del sentimento standard risiede nel livello di dettaglio dei dati di output. L'analisi del sentimento standard determina il sentimento predominante per ogni documento di input, ma non offre informazioni per analisi più approfondite. D'altra parte, l'analisi mirata del sentimento determina il sentimento per entità specifiche presenti in ogni documento di input. Questo consente di analizzare i dati di output per individuare prodotti e servizi specifici che ricevono feedback positivi o negativi (Figura 2.11 e 2.12). Ad esempio, consideriamo una serie di recensioni su ristoranti. Un cliente scrive: "I tacos erano deliziosi e il personale era cordiale". Mediante l'analisi del sentimento mirato, è possibile determinare non solo il sentimento generale della recensione, ma anche il sentimento associato ai tacos e al personale, offrendo così una visione più dettagliata e focalizzata dei feedback dei clienti.

  Il sentimento mirato fornisce i seguenti risultati per ogni lavoro di analisi:

  – Le entità menzionate nei documenti.
  – Il tipo di entità per ogni entità menzionata.
  – Il sentimento e un punteggio per ciascuna menzione dell'entità. Questo può essere:
    * *Positivo:* l'entità menzionata esprime un sentimento positivo.
    * *Negativo:* la menzione dell'entità esprime un sentimento negativo.
    * *Mista:* la menzione dell'entità esprime sentimenti sia positivi che negativi.
    * *Neutro:* la menzione dell'entità non esprime sentimenti positivi o negativi.
  – Gruppi di menzioni (gruppi di co-riferimento) che corrispondono a una singola entità.

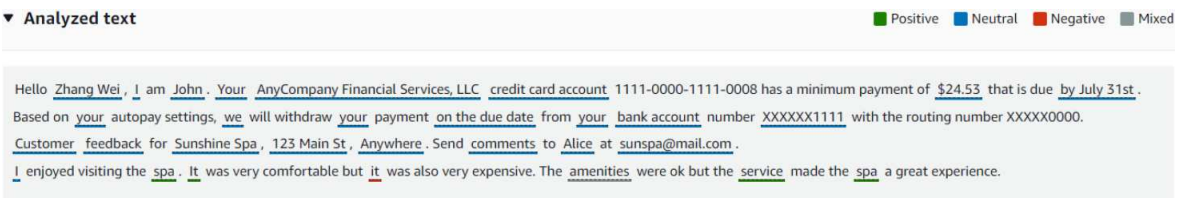


Figura 2.11: Rilevamento delle entità e relativo sentimento nel testo

- *Analisi della sintassi:* l'analisi della sintassi in Amazon Comprehend è un'operazione che consente di analizzare la struttura grammaticale di un testo per identificare le relazioni

Entity	Entity type	Entity score	Primary sentiment	Positive score	Negative score	Neutral score
Zhang Wei (5)	PERSON	-	NEUTRAL	-	-	-
John (3)	PERSON	-	NEUTRAL	-	-	-
AnyCompany Financial Services, LLC (2)	ORGANIZATION	-	NEUTRAL	-	-	-
we	ORGANIZATION	0.99+	NEUTRAL	0	0	0.99+
AnyCompany Financial Services, LLC	ORGANIZATION	0.99+	NEUTRAL	0.00	0.00	0.99+
credit card account	OTHER	0.99+	NEUTRAL	0.00	0.00	0.99+
$24.53	QUANTITY	0.99+	NEUTRAL	0.00	0.00	0.99+
on the due date (2)	DATE	-	NEUTRAL	-	-	-
on the due date	DATE	0.99+	NEUTRAL	0.00	0.00	0.99+
by July 31st	DATE	0.99+	NEUTRAL	0.00	0.00	0.99+
bank account	OTHER	0.99+	NEUTRAL	0.00	0.00	0.99+
XXXXXX1111	OTHER	0.73	NEUTRAL	0.00	0.00	0.99+
Customer	PERSON	0.81	NEUTRAL	0	0	1.00
feedback	OTHER	0.68	NEUTRAL	0	0	1.00

Figura 2.12: Risultati dell'analisi del sentimento mirato

sintattiche tra le parole e le frasi ivi presenti.
Il suo funzionamento si basa su quattro passaggi fondamentali:

- *Tokenizzazione:* il testo viene suddiviso in token, ovvero unità linguistiche come parole e punteggiatura.
- *Analisi delle dipendenze:* Comprehend analizza le relazioni di dipendenza tra i token all'interno di una frase. Questo include l'individuazione dei soggetti, dei verbi, degli oggetti e di altre parti del discorso, nonché la determinazione delle relazioni di dipendenza tra di esse.
- *Generazione dell'albero sintattico:* basandosi sull'analisi delle dipendenze, Comprehend genera un albero sintattico che rappresenta la struttura grammaticale del testo. Questo albero mostra come le parole sono connesse tra loro e fornisce una visualizzazione della struttura sintattica del documento.
- *Identificazione delle entità:* Comprehend può anche identificare entità rilevanti nel testo, come nomi di persone, luoghi, date, quantità, e altro ancora.

Word	Part of speech	Confidence
Hello	Proper noun	0.36
Zhang	Proper noun	1.00
Wei	Proper noun	1.00
,	Punctuation	1.00
I	Pronoun	0.99+
am	Verb	0.99+
John	Proper noun	0.99+
.	Punctuation	1.00
Your	Pronoun	1.00
AnyCompany	Proper noun	0.99+

Figura 2.13: Risultati dell'analisi della sintassi

Vantaggi di Amazon Comprehend

I vantaggi di Amazon Comprehend sono i seguenti:

- Rimuove la complessità legata all'integrazione di funzionalità di analisi del testo nelle applicazioni, rendendo disponibile un'elaborazione del linguaggio naturale con una semplice API[2], senza la necessità di esperienza nell'analisi testuale.
- I modelli vengono costantemente addestrati con nuovi dati su più domini per migliorare la precisione.
- Consente di analizzare milioni di documenti per scoprire le informazioni in essi contenute.
- È progettato per funzionare perfettamente con altri servizi AWS.
- Permette di crittografare i risultati di output e dei dati di volume.

2.4.3 Amazon Rekognition

Amazon Rekognition è un servizio di analisi di immagini e video che sfrutta l'Intelligenza Artificiale per identificare, analizzare e interpretare automaticamente contenuti visivi. Mediante l'uso di tecnologie avanzate di machine learning, Rekognition è in grado di riconoscere oggetti, volti (Figura 2.14), testo e scene nelle immagini e nei video in modo automatizzato e preciso.

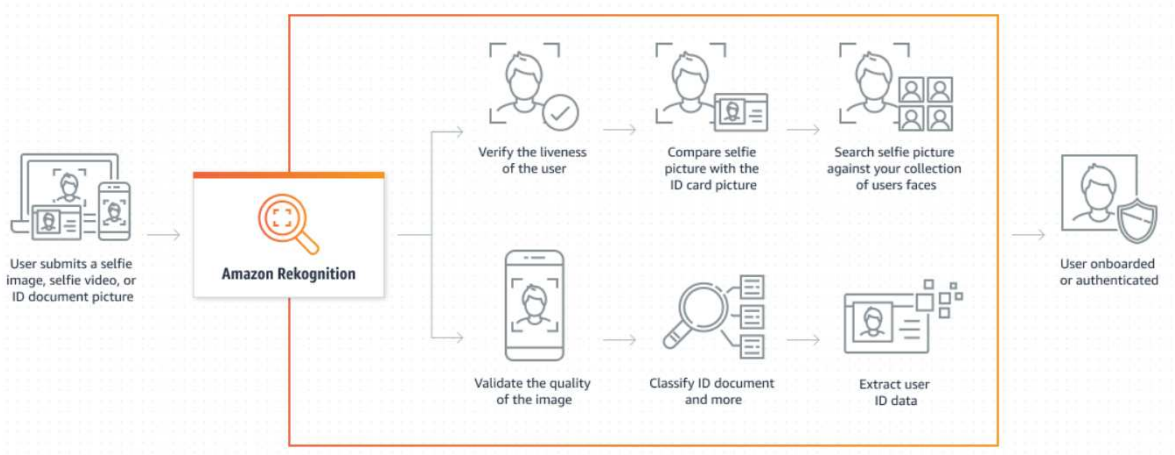


Figura 2.14: Schema di funzionamento del rilevamento volti per l'autenticazione

Funzionalità di analisi delle immagini di Amazon Rekognition

Le principali funzionalità di analisi delle immagini fornite da Amazon Rekognition sono le seguenti:

- *Rilevamento delle etichette:* un'etichetta, o tag, è un oggetto o un concetto (incluse scene e azioni) che si trova in un'immagine o in un video e che ne descrive il suo contenuto. Con Rekognition è possibile identificare etichette e scene nelle immagini in base alle esigenze (Figura 2.15).

[2]Un'API, o interfaccia di programmazione delle applicazioni, rappresenta un insieme di regole stabilite che facilita la comunicazione tra diverse applicazioni. Questo strumento agisce come uno strato intermedio per gestire lo scambio di dati tra i sistemi, consentendo alle aziende di esporre dati e funzionalità delle applicazioni all'esterno. Ciò li rende accessibili a sviluppatori di terze parti, partner commerciali e vari dipartimenti aziendali.

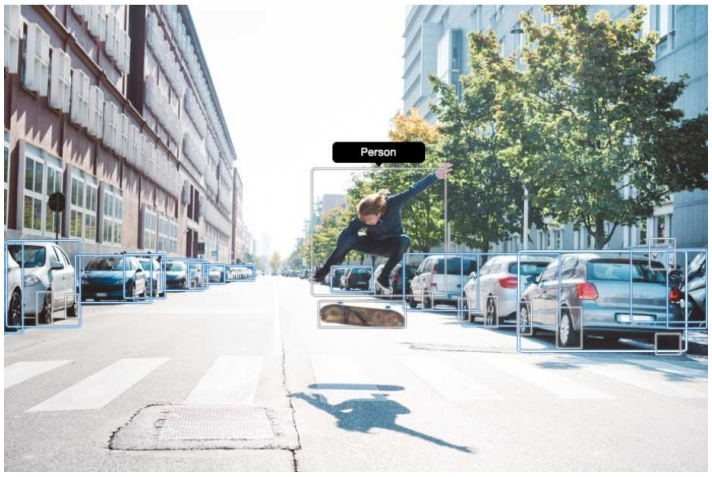


Figura 2.15: Risultati del rilevamento delle etichette

- *Proprietà dell'immagine*: fornisce dati sulla qualità dell'immagine, compresi nitidezza, luminosità e contrasto per l'intera immagine. La nitidezza e la luminosità sono valutate sia per il primo piano che per lo sfondo dell'immagine. Inoltre, le proprietà dell'immagine consentono di individuare i colori predominanti per l'intera immagine, il primo piano, lo sfondo e gli oggetti con riquadri di delimitazione (Figura 2.16).

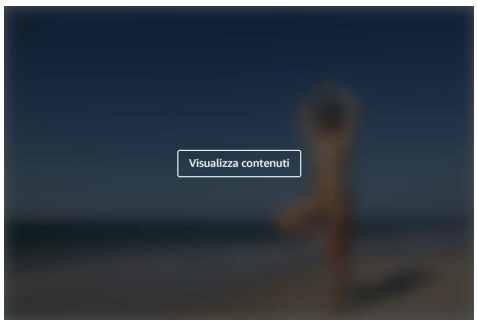
(a) Immagine di riferimento

Image Properties	Dominant Colors Examples and Pixel Percentage	Image Quality
Entire Image	Hex code #808080, RGB (128, 128, 128), 15.72 Hex code #000000, RGB (0, 0, 0), 15.10 Hex code #696969, RGB (105, 105, 105), 14.02 Hex code #8fbc8f, RGB (143, 188, 143), 12.70 Hex code #5f9ea0, RGB (95, 158, 160), 11.92	Brightness: 76.08 Sharpness: 89.72 Contrast: 88.42
Foreground	Hex code #8fbc8f, RGB (143, 188, 143), 30.18 Hex code #5f9ea0, RGB (95, 158, 160), 24.29 Hex code #000000, RGB (0, 0, 0), 12.02 Hex code #2f4f4f, RGB (47, 79, 79), 9.20 Hex code #696969, RGB (105, 105, 105), 8.95	Brightness: 79.48 Sharpness: 93.47
Background	Hex code #808080, RGB (128, 128, 128), 21.16 Hex code #2f4f4f, RGB (47, 79, 79), 14.61 Hex code #000000, RGB (0, 0, 0), 14.23 Hex code #696969, RGB (105, 105, 105), 13.19 Hex code #ffebcd, RGB (255, 235, 205), 12.80	Brightness: 74.42 Sharpness: 87.84
Car (example of objects with bounding boxes)	Hex code #5f9ea0, RGB (95, 158, 160), 29.18 Hex code #8fbc8f, RGB (143, 188, 143), 14.39 Hex code #000000, RGB (0, 0, 0), 11.76 Hex code #808080, RGB (128, 128, 128), 11.38 Hex code #2f4f4f, RGB (47, 79, 79), 9.44	Not applicable

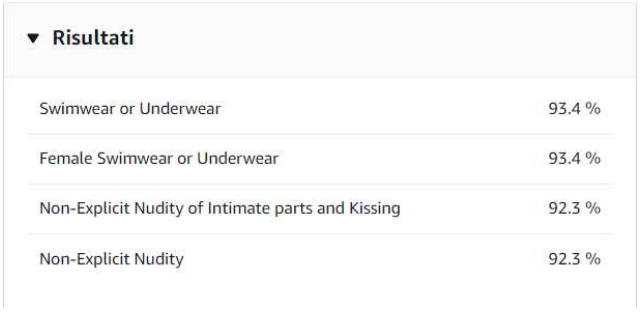
(b) Risultati del rilevamento delle proprietà dell'immagine

Figura 2.16: Rilevamento delle proprietà dell'immagine

- *Rilevamento di contenuti non sicuri:* questa funzionalità è in grado di rilevare contenuti inappropriati, indesiderati o offensivi. L'API Content Moderation restituisce, anche, un elenco gerarchico di etichette rilevate che indicano categorie specifiche di contenuti non sicuri, il che consente di filtrare concetti specifici e gestire grandi volumi di contenuti generati dagli utenti (Figura 2.17).



(a) Immagine di riferimento



(b) Risultati della moderazione dell'immagine

Figura 2.17: Rilevamento di contenuti non sicuri

- *Rilevamento e analisi del viso:* questa funzionalità è in grado di rilevare e analizzare diversi componenti e attributi del viso, come espressioni emotive (felicità, tristezza o sorpresa), informazioni demografiche (come sesso o età), occlusione facciale (quando gli occhi, il naso e/o la bocca di un viso sono bloccati da occhiali da sole scuri, maschere, mani, etc.) e direzione dello sguardo (definita da altezza e imbardata Figura 2.19). Basandosi su questa funzionalità, Amazon Rekognition permette di effettuare altre operazioni come:
  - *Confronto facciale*, dove si confrontano i volti per vedere la corrispondenza tra loro in base a una percentuale di somiglianza (Figura 2.18).

(a) Risultati del primo confronto

(b) Risultati del secondo confronto

Figura 2.18: Confronto facciale

  - *Riconoscimento dei volti*, con cui è possibile cercare nelle immagini, nonché nei video archiviati e in streaming, i volti che corrispondono a quelli presenti in container noti come raccolte di volti.
  - *Face Liveness Detection*, cioè una funzionalità di machine learning progettata per aiutare gli sviluppatori a scoraggiare le frodi nella verifica dell'identità basata sul volto. La funzionalità consente di verificare che un utente sia fisicamente presente davanti alla telecamera e non sia un cattivo attore a falsificare il volto dell'utente.
  - *Verifica utente basata sul volto*, con cui è possibile confermare l'identità dell'utente confrontandone l'immagine dal vivo con un'immagine di riferimento.

- *Riconoscimento di volti celebri:* può identificare le celebrità nelle immagini fornite e nei video. Amazon Rekognition è in grado di riconoscere migliaia di volti celebri in un

looks like a face	99.9 %
appears to be female	99.9 %
age range	26 - 34 years old
smiling	62.5 %
appears to be happy	92.1 %
wearing glasses	100 %
wearing sunglasses	100 %
eyes are open	98.7 %
mouth is open	94.8 %
does not have a mustache	99.9 %
does not have a beard	89.5 %
face is occluded	99.9 %

(a) Immagine di riferimento **(b)** Risultati dell'analisi facciale

Figura 2.19: Rilevamento e analisi del viso

gran numero di categorie, ad esempio politica, sport, affari, intrattenimento e media (Figura 2.20).

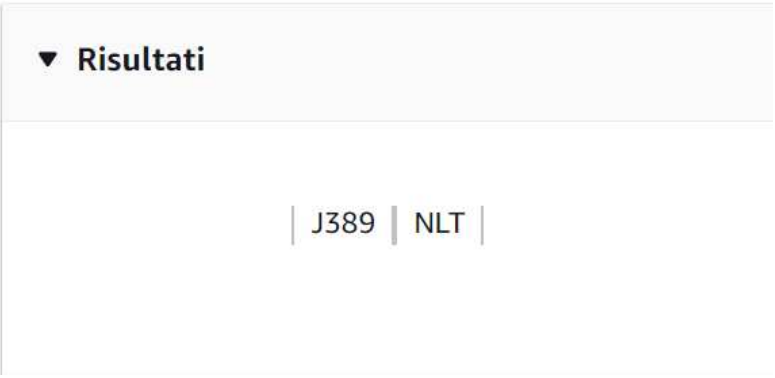
(a) Immagine di riferimento **(b)** Risultati del riconoscimento

Figura 2.20: Riconoscimento di volti celebri

- *Rilevamento del testo:* la funzione Text (Testo) in Image (Immagine) di Amazon Rekognition permette di individuare ed estrapolare contenuti di testo dalle immagini (Figura 2.21).
- *Rilevamento di dispositivi di protezione individuale (PPE):* questa funzione rileva nelle immagini i dispositivi di protezione individuale, come coperture per il viso, copricapi e coperture per le mani sulle persone (Figura 2.22). È possibile utilizzare il rilevamento dei PPE laddove la sicurezza ha la massima priorità.

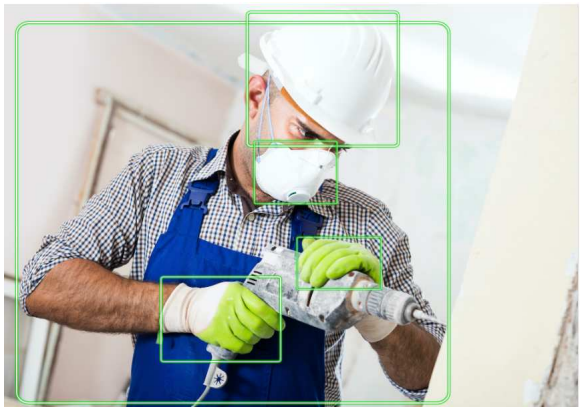
Funzionalità di analisi dei video di Amazon Rekognition

Le funzionalità di analisi dei video eseguono le stesse operazioni di individuazione delle etichette viste nell'analisi di immagini statiche. La principale differenza risiede nel tipo di contenuto video che viene analizzato. In particolare, esistono le seguenti due possibilità:



(a) Immagine di riferimento **(b)** Risultati del rilevamento del testo nell'immagine

Figura 2.21: Rilevamento del testo nell'immagine



Person detected	99.9 %
Face detected	99.9 %
Face cover detected	99.8 %
Face cover on nose : true	98.5 %
Left hand detected	96.8 %
Hand cover detected on left hand	86.9 %
Hand cover on left hand : true	96.2 %
Right hand detected	99.8 %
Hand cover detected on right hand	98.2 %
Hand cover on right hand : true	99.8 %
Head detected	99.9 %
Head cover detected	86.1 %
Head cover on head : true	99.9 %

(a) Immagine di riferimento **(b)** Risultati del riconoscimento dei PPE

Figura 2.22: Rilevamento di dispositivi di protezione individuale

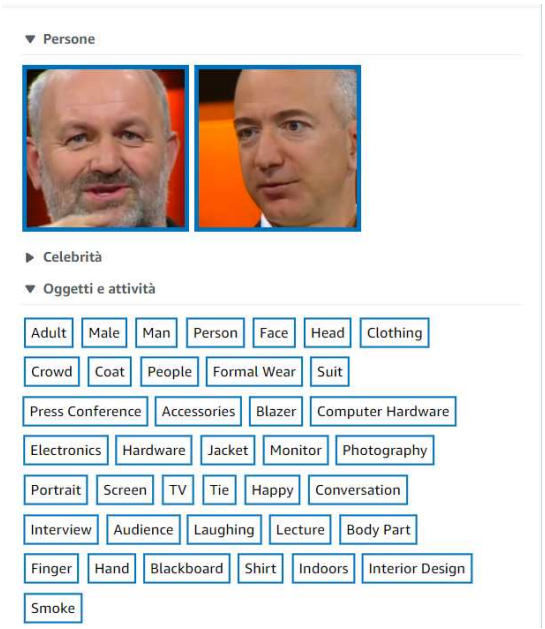
- *Analisi di video archiviati.* Nella Figura 2.23 è riportato il funzionamento di questo tipo di analisi video.
- *Analisi in tempo reale di streaming video.* Nella Figura 2.24 è riportato il funzionamento dell'analisi in tempo reale.

Vantaggi di Amazon Rekognition

I principali punti di forza di Amazon Rekognition sono i seguenti:

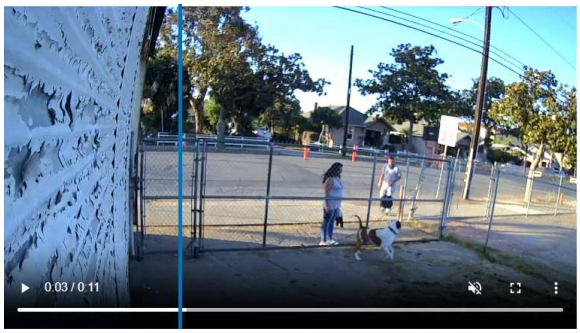
- *Integrazione di potenti analisi di immagini e video senza aver bisogno di esperienza in deep learning*; con l'API è possibile integrare analisi di immagini e video in qualsiasi applicazione Web, mobile o connessa.

(a) Frame di riferimento

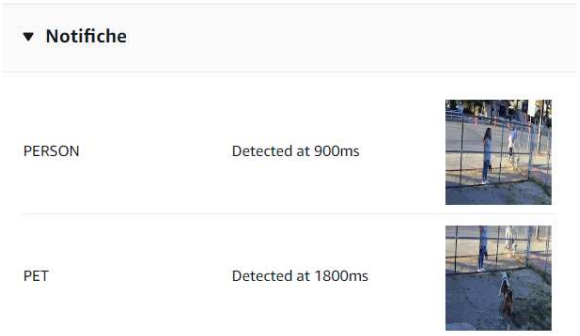


(b) Risultati dell'analisi

Figura 2.23: Analisi di video archiviati



(a) Frame di riferimento



(b) Risultati dell'analisi

Figura 2.24: Analisi in tempo reale di streaming video

- *Analisi scalabile delle immagini*; Amazon Rekognition consente di analizzare milioni di immagini per curare e organizzare enormi volumi di dati visivi.
- *Integrazione completa* con altri servizi AWS.
- *Personalizzazione semplice*; grazie alle proprietà di Amazon Rekognition è possibile migliorare l'accuratezza della classificazione e del rilevamento degli oggetti creando adattatori addestrati sui dati forniti.

2.4.4 Amazon Translate

Amazon Translate è un servizio di traduzione di testi che utilizza tecnologie avanzate di machine learning per fornire traduzioni di alta qualità. Esso è utile per tradurre documenti di testo non strutturati o per creare applicazioni che funzionano in più lingue (Figura 2.25).

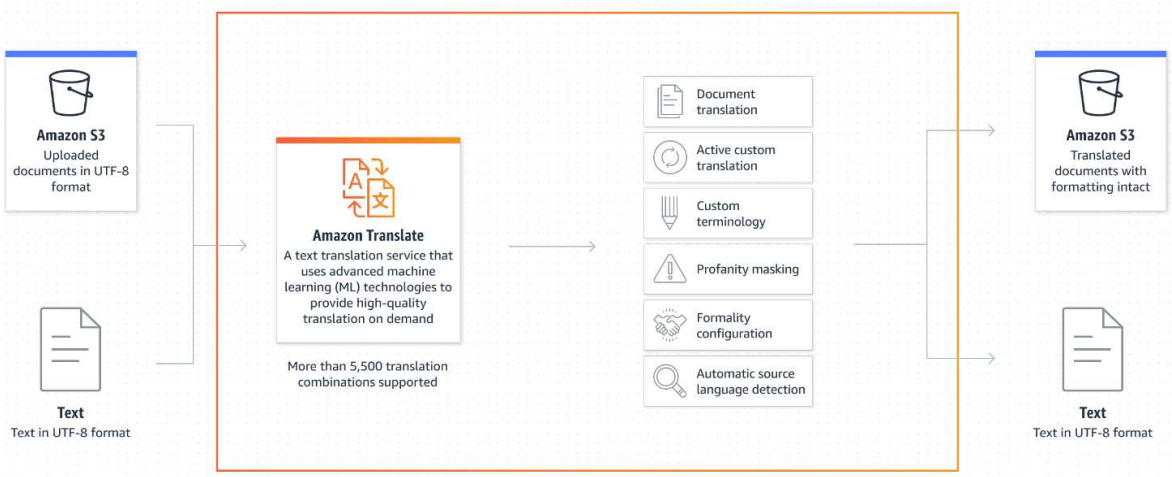


Figura 2.25: Schema generale di Amazon Translate

Funzionalità di Amazon Translate

Amazon Translate non consente semplicemente di tradurre documenti o testi ma ci permette di personalizzare tale traduzione attraverso quattro funzionalità aggiuntive:

- *Terminologia personalizzata:* permette, oltre alle normali richieste di traduzione, di assicurare che i nomi dei marchi, dei personaggi, dei modelli e altri contenuti esclusivi vengano tradotti nel risultato desiderato.
- *Brevità:* la brevità riduce la lunghezza dell'output per la maggior parte delle traduzioni. Questo perché, a volte, l'output della traduzione è più lungo (in termini di numero di caratteri) di quanto desiderato, e ciò può causare problemi in alcuni scenari (ad esempio didascalie, sottotitoli, titoli o campi modulo) quando se non c'è spazio per caratteri aggiuntivi.
- *Formalità:* permette di specificare facoltativamente il livello di formalità desiderato (formale o informale) per le traduzioni nelle lingue di destinazione supportate.
- *Mascheramento del contenuto volgare:* consente di mascherare parole e frasi profane nell'output della traduzione. Può succedere, però, che, in alcuni casi, una parola volgare nell'input di origine potrebbe naturalmente diventare inoffensiva nell'output tradotto. In questi casi, non viene applicata alcuna mascheratura.

Vantaggi di Amazon Translate

I principali punti di forza di Amazon Translate sono i seguenti:

- Offre traduzioni precise e di alta qualità mediante modelli di machine learning addestrati su vasti dataset multilingue.
- Offre supporto per numerose lingue, consentendo di tradurre tra una vasta gamma di combinazioni linguistiche.
- Facilità l'integrazione grazie alle API fornite.
- Garantisce una disponibilità elevata e una scalabilità dinamica del servizio, consentendo di gestire traduzioni su larga scala in modo efficiente.
- Protegge la riservatezza e l'integrità dei dati durante il processo di traduzione, garantendo la conformità con le normative sulla privacy dei dati.

2.4.5 Amazon API Gateway

Amazon API Gateway è un servizio che semplifica la creazione, la pubblicazione, la manutenzione, il monitoraggio e la protezione di API REST, HTTP e WebSocket per le applicazioni. Si tratta di uno strumento essenziale per la creazione di microservizi e architetture serverless, consentendo agli sviluppatori di esporre facilmente le proprie risorse e funzionalità tramite API web.

Architettura di API Gateway

Il diagramma in Figura 2.26 illustra come le API offrano un'esperienza di sviluppo integrata e coerente per la creazione di applicazioni AWS senza server. API Gateway gestisce tutte le attività di accettazione ed elaborazione relative a molteplici chiamate API simultanee. Tali attività includono la gestione del traffico, il controllo dell'autorizzazione e dell'accesso, il monitoraggio e la gestione delle versioni delle API.

API Gateway funge da "porta principale" attraverso cui le applicazioni possono accedere ai dati, alla logica di business o alle funzionalità dei servizi di back-end, semplificando l'accesso ai servizi e alle risorse AWS e fornendo un'interfaccia unificata e scalabile per lo sviluppo e la gestione delle API.

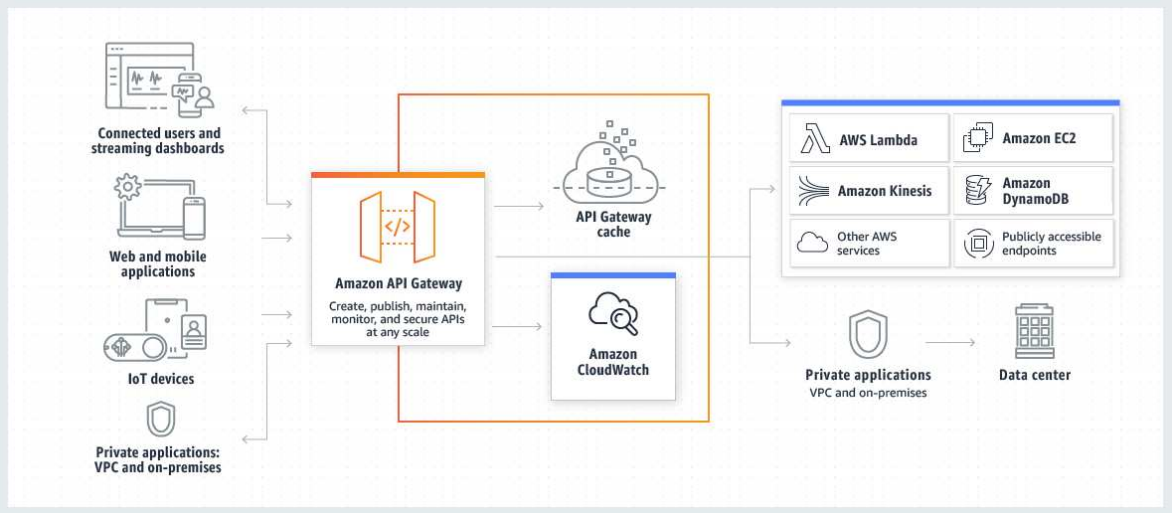


Figura 2.26: Schema dell'architettura di API Gateway

Casi d'uso di Amazon API Gateway

I principali casi d'uso di Amazon API Gateway sono i seguenti:

- *API REST:* acronimo di Application Programming Interface Representational State Transfer; rappresenta un tipo di architettura per la progettazione di servizi web che consentono ai client di accedere e manipolare le risorse su un server, solitamente rappresentate da URL univoci, utilizzando un set predefinito di operazioni standard, come GET, POST, PUT e DELETE, tramite il protocollo HTTP.
  Consideriamo un esempio: "/incomes" potrebbe essere il percorso di una risorsa che rappresenta il reddito di un'utente. Tale risorsa può avere più operazioni definite tra quelle citate in precedenza, e la combinazione tra percorso della risorsa e operazione identifica un metodo dell'API. Quindi per esempio il metodo "POST /incomes" potrebbe aggiungere il nuovo reddito dell'utente oppure il metodo "GET /incomes" potrebbe ricavare le spese sostenute da esso.

- *API HTTP:* consentono di creare API RESTful, caratterizzate da una latenza minore e costi inferiori rispetto alle API REST. Queste ultime supportano più funzionalità; invece, le API HTTP sono progettate con funzionalità minime in modo che possano essere offerte a un prezzo inferiore.
- *API WebSocket:* il client e il server possono entrambi scambiarsi messaggi in qualsiasi momento. I server di back-end possono eseguire facilmente il push dei dati agli utenti e ai dispositivi connessi, evitando la necessità di implementare complessi meccanismi di polling[3]. Le API WebSocket sono utili per creare applicazioni per comunicazioni sicure e in tempo reale, senza dover effettuare il provisioning o la gestione dei server per gestire connessioni o scambi di dati su larga scala. I casi d'uso di riferimento includono applicazioni in tempo reale come le seguenti:
  - chat;
  - pannelli di controllo in tempo reale, ad esempio ticker azionari;
  - avvisi e notifiche in tempo reale.

Concetti di API Gateway

I principali concetti alla base di API Gateway sono i seguenti:

- *Distribuzione API:* è uno snapshot "point-in-time" dell'API; si riferisce a un istante temporale specifico in cui viene catturato lo stato corrente dell'API e di tutte le sue risorse correlate. Questo snapshot registra la configurazione dell'API, inclusi endpoint, metodi, autorizzazioni, mapping di integrazione e altre impostazioni di configurazione, proprio come si trovano in quel momento preciso.
- *API endpoint:* è un punto di accesso attraverso il quale le applicazioni, i servizi o i dispositivi possono interagire con un'API. In termini più semplici, un endpoint API è un URL specifico che rappresenta una risorsa su un server.
- *Fase API:* è un riferimento logico a uno stato del ciclo di vita dell'API.
- *URL di callback:* è uno strumento per inviare messaggi al client dal sistema di back-end quando si stabilisce una nuova connessione WebSocket.
- *Route:* viene utilizzata per indirizzare messaggi in entrata a un'integrazione specifica, ad esempio una funzione AWS Lambda, in base al contenuto dei messaggi stessi. Quando si crea un'API WebSocket si specifica una chiave route e un back-end di integrazione. Quando viene trovata una corrispondenza per la chiave route in un messaggio in entrata, viene richiamato il back-end di integrazione.
- *Richiesta di instradamento:* è un'interfaccia pubblica di un metodo API WebSocket che definisce il corpo da inviare nelle richieste per accedere al back-end tramite l'API.
- *Rispota di instradamento:* è un'interfaccia pubblica di un'API WebSocket che definisce i codici di stato, le intestazioni e i modelli di corpo che ci si aspetta da una risposta API Gateway.
- *Connessione WebSocket:* API Gateway mantiene una connessione permanente tra i client e se stesso. Non esiste alcuna connessione permanente tra API Gateway e le integrazioni di back-end, poiché questi servizi vengono richiamati in base alle esigenze, a seconda del contenuto dei messaggi ricevuti dai client.

[3] I meccanismi di polling sono utilizzati per ottenere aggiornamenti o informazioni da una risorsa o da un sistema in modo periodico, controllando periodicamente se sono disponibili nuovi dati o stati.

- *Richiesta di integrazione:* è un'interfaccia interna di una route API WebSocket o di un metodo API REST che gestisce la comunicazione tra le richieste in arrivo e il back-end del sistema, garantendo che i dati siano trasferiti in modo corretto e comprensibile per il sistema che deve elaborarli.
- *Risposta di integrazione:* è un'interfaccia interna di una route API WebSocket o di un metodo API REST, in cui i codici di stato, le intestazioni e il payload ricevuti dal back-end vengono mappati nel formato di risposta restituito a un'app client.
- *Modello di mappatura:* è uno script che trasforma il corpo di una richiesta dal formato dati del front-end al formato dati del back-end e la risposta dal formato dati del back-end al formato dati del front-end. I modelli di mappatura possono essere specificati nella richiesta o nella risposta di integrazione.
- *Richiesta metodo:* è un'interfaccia pubblica di un metodo API che definisce i parametri e il corpo da inviare nelle richieste per accedere al back-end tramite l'API.
- *Risposta metodo:* è un'interfaccia pubblica di un'API REST che definisce i codici di stato, le intestazioni e i modelli di corpo, che ci si aspetta in risposta dall'API.
- *Modello:*è uno schema di dati che specifica la struttura di dati di una richiesta o di un payload di risposta; viene anche utilizzato per convalidare i payload.
- *Integrazione fittizia:* le risposte API vengono generate direttamente da API Gateway senza che sia richiesto un back-end di integrazione. A tale scopo, è necessario configurare la richiesta e la risposta di integrazione del metodo in maniera opportuna.
- *Integrazione privata:* consente a un client di accedere alle risorse all'interno del VPC (Virtual Private Cloud) di un cliente tramite un endpoint API REST privato senza esporre le risorse alla rete Internet pubblica.
- *Integrazione proxy:* è una configurazione di un'integrazione API Gateway semplificata che consente di instradare le richieste HTTP in arrivo direttamente al back-end senza la necessità di definire esplicitamente risorse e metodi per ogni singola richiesta. Questo tipo di configurazione è utile quando si desidera gestire dinamicamente molte risorse o metodi con un'unica configurazione.
  Ci sono due tipi di integrazione proxy supportate da API Gateway:
  - *Integrazione Proxy Lambda:* la richiesta HTTP completa viene inviata come input alla funzione Lambda di back-end. Quest'ultima elabora la richiesta e restituisce una risposta che API Gateway trasforma, poi, in una risposta HTTP per il front-end.
  - *Integrazione Proxy HTTP:* trasferisce la richiesta HTTP completa dal front-end al back-end nonché la risposta completa dal back-end al front-end. Ciò significa che API Gateway agisce come un intermediario trasparente che inoltra le richieste e le risposte tra il client e il server HTTP di back-end.

2.4.6 Amazon S3

Amazon Simple Storage Service (Amazon S3) rappresenta una soluzione di archiviazione di oggetti all'avanguardia, caratterizzata da scalabilità, affidabilità dei dati, sicurezza e performance. Questo servizio fornisce la possibilità di memorizzare e proteggere qualsiasi volume di dati in modo efficiente. Le sue applicazioni spaziano in diversi ambiti, inclusi i data lake, la gestione di siti web, le app mobili, i processi di backup e ripristino, gli archivi, le soluzioni aziendali, i dispositivi IoT e le attività di analisi dei Big Data (Figura 2.27).

Inoltre, Amazon S3 mette a disposizione funzionalità di gestione avanzate che consentono di ottimizzare, organizzare e controllare l'accesso ai dati, rispondendo, così, a esigenze specifiche in termini di requisiti aziendali, organizzativi e di conformità normativa.

Figura 2.27: Schema generale di Amazon S3

Fuzionamento e concetti di Amazon S3

Amazon S3 è un servizio di archiviazione che consente di memorizzare dati sotto forma di oggetti, identificati da una chiave, all'interno di contenitori chiamati "bucket" (Figura 2.28).

Questi ultimi costituiscono i contenitori per gli oggetti archiviati e possono ospitare un numero illimitato di oggetti. Gli oggetti, a loro volta, rappresentano le unità fondamentali di archiviazione e sono composti da dati e metadati. Questi ultimi consistono in un insieme di coppie nome-valore che descrivono l'oggetto e possono includere dati come la data dell'ultima modifica, e metadati standard HTTP, come Content-Type. È anche possibile definire metadati personalizzati al momento dell'archiviazione dell'oggetto.

La chiave dell'oggetto, o nome della chiave, è l'identificatore unico di un oggetto all'interno di un bucket. La combinazione di bucket, chiave dell'oggetto ed, eventualmente, ID della versione assicura un'identificazione univoca per ciascun oggetto. Questo concetto sottolinea la struttura fondamentale di Amazon S3 come una mappatura tra "bucket + chiave dell'oggetto + versione" e l'oggetto stesso.

Il controllo delle versioni S3 consente di conservare più versioni di un oggetto all'interno dello stesso bucket. Grazie a questo servizio è possibile archiviare, recuperare e ripristinare qualsiasi versione di un oggetto memorizzato nei bucket. Il controllo delle versioni S3 facilita il ripristino in caso di errori di applicazione oppure operazioni non intenzionali da parte dell'utente, offrendo la possibilità di gestire facilmente le variazioni e le modifiche agli oggetti nel tempo.

Nome	Tipo	Ultima modifica	Dimensioni	Classe di storage
f82490d9-8be7-4a6b-aba1-a2a8268384f5.jpg	jpg	22 Dec 2023 10:07:40 AM CET	190.4 KB	Standard
f3acf9f5-d898-48bb-af76-f7d0c6885942.jpg	jpg	22 Dec 2023 10:13:50 AM CET	184.7 KB	Standard
b51a0958-e2ce-4fa7-846d-df13b9bc6c09.jpg	jpg	22 Dec 2023 10:07:15 AM CET	191.2 KB	Standard
5c90b907-84d9-4181-83fd-219834c59ceb.jpg	jpg	22 Dec 2023 10:07:30 AM CET	191.0 KB	Standard
110e8f03-8c87-4138-8bb7-c92f6e2cc2eb.jpg	jpg	22 Dec 2023 10:06:52 AM CET	190.2 KB	Standard

Figura 2.28: Esempio di un bucket che contiene immagini

Caratteristiche di Amazon S3

Amazon S3 offre una vasta gamma di classi di storage progettate per soddisfare una varietà di casi d'uso. Per esempio, è possibile archiviare dati critici per la produzione su S3 Standard o S3 Express One Zone per un accesso più frequente; è anche possibile risparmiare sui costi archiviando dati raramente accessibili su S3 Standard-IA o S3 One Zone-IA; infine, è possibile conservare i dati a un costo inferiore su S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval e S3 Glacier Deep Archive (Figura 2.29).

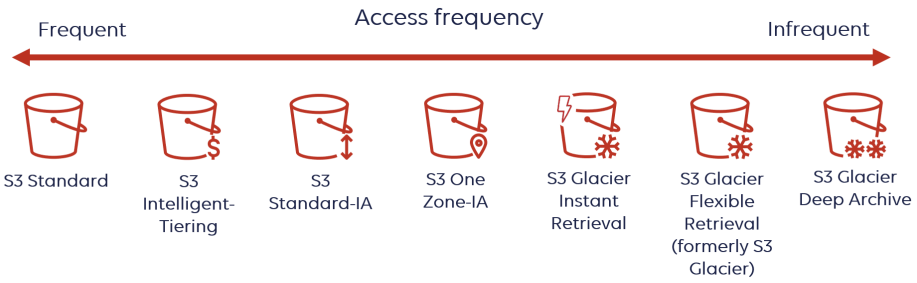


Figura 2.29: I vari tipi di classi di archiviazione di Amazon S3

S3 Express One Zone di Amazon è una classe di archiviazione a zona singola ad alte prestazioni, progettata appositamente per garantire un accesso coerente ai dati con una latenza di pochi millisecondi per le applicazioni sensibili alla velocità. È caratterizzata da velocità di accesso ai dati fino a 10 volte superiori e costi di richiesta inferiori rispetto a S3 Standard. Inoltre, è la prima classe di archiviazione S3 che consente la selezione di una singola zona di disponibilità, consentendo la co-locazione di archiviazione di oggetti e risorse di calcolo per un accesso ottimale. I dati vengono memorizzati in un nuovo tipo di bucket, denominato bucket di directory Amazon S3, per aumentare ulteriormente la velocità di accesso e supportare centinaia di migliaia di richieste al secondo.

Figura 2.30: Schema di Amazon S3 Express One Zone

D'altra parte, per dati con modelli di accesso mutevoli o sconosciuti, S3 Intelligent-Tiering offre un'opzione ottimale. Questa classe di archiviazione ottimizza i costi spostando automaticamente i dati tra quattro livelli di accesso in base ai modelli di accesso. I due livelli a bassa latenza sono ottimizzati per l'accesso frequente e sporadico, mentre i due livelli di archivio sono progettati per l'accesso asincrono e per i dati a cui si accede raramente.

Amazon S3 offre, inoltre, robuste funzionalità di gestione dello storage che consentono di controllare i costi, soddisfare i requisiti normativi, ridurre la latenza e garantire la conformità. Amazon S3 possiede i seguenti punti di forza:

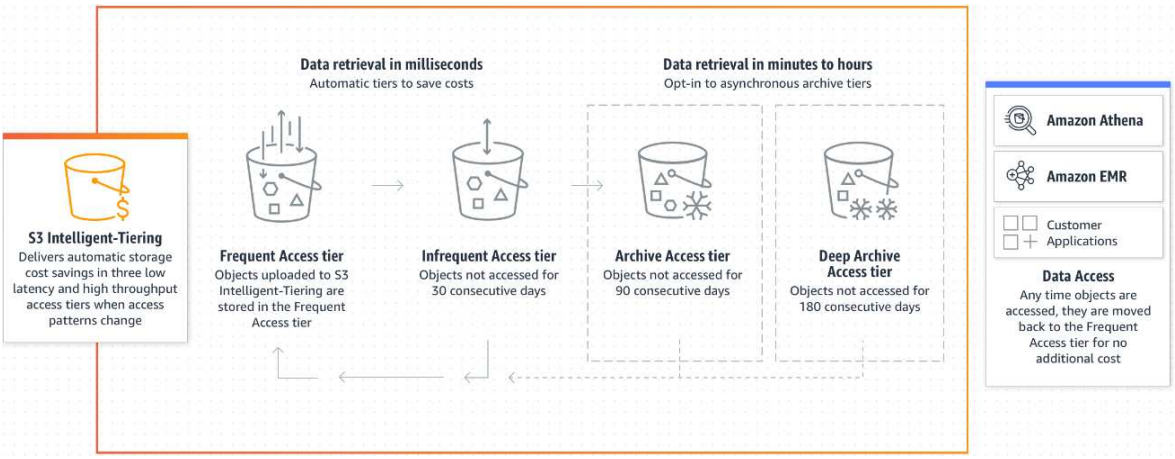


Figura 2.31: Schema di Amazon S3 Intelligent-Tiering

- Il Ciclo di Vita di S3 permette di configurare le fasi del ciclo di vita degli oggetti per ottimizzare l'efficienza dei costi durante tutto il loro percorso. È possibile spostare gli oggetti tra diverse classi di archiviazione S3 o eliminare oggetti giunti alla fine del loro ciclo.
- Il Blocco degli Oggetti S3 impedisce l'eliminazione o la sovrascrittura di un oggetto per un periodo definito o indefinito. Questa funzionalità è ideale per soddisfare requisiti normativi che richiedono archiviazione write-once-read-many (WORM) o per aggiungere ulteriori strati di protezione contro modifiche o cancellazioni accidentali.
- La Replica di S3 permette di replicare gli oggetti e i relativi metadati e tag in uno o più bucket di destinazione, sia nella stessa regione AWS che in regioni diverse. Ciò aiuta a ridurre la latenza, garantire la conformità e rafforzare la sicurezza dei dati.
- Le Operazioni in Batch di S3 consentono di gestire un grande numero di oggetti con una singola richiesta API S3. Questa funzionalità offre la flessibilità di eseguire operazioni come Copia, Invocazione di funzioni AWS Lambda e Ripristino su milioni o miliardi di oggetti in modo efficiente e scalabile.

2.4.7 AWS Lambda

AWS Lambda è un servizio di elaborazione serverless che consente agli sviluppatori di eseguire codice senza dover gestire direttamente i server sottostanti. Con Lambda è possibile eseguire il proprio codice su un'infrastruttura di elaborazione ad alta disponibilità senza dover effettuare il provisioning o la gestione dei server.

Il codice viene organizzato in funzioni Lambda, e Lambda esegue la funzione solo quando necessario, scalando automaticamente in base al carico di lavoro. Ciò significa che non è necessario preoccuparsi della manutenzione del server, del provisioning delle risorse o della scalabilità della capacità.

Lambda è un servizio di calcolo ideale in situazioni in cui il carico di lavoro può variare notevolmente nel tempo, con picchi di attività seguiti da periodi di inattività, come per esempio:

- *Elaborazione di file:* Lambda viene utilizzato con Amazon Simple Storage Service (Amazon S3) per avviare l'elaborazione dei dati in tempo reale dopo un caricamento di oggetti in un bucket (Figura 2.32).

- *Elaborazione in streaming:* Lambda viene usato in combinazione con Amazon Kinesis per elaborare dati in streaming in tempo reale per il monitoraggio delle attività delle applicazioni, l'elaborazione degli ordini delle transazioni, la telemetria dei dati dei dispositivi Internet delle cose (IoT) e molto altro.
- *Applicazioni Web:* Lambda può essere accoppiato con altri servizi AWS per creare applicazioni Web scalabili.
- *Back-end IoT:* è possibile creare back-end serverless utilizzando Lambda per gestire richieste di API Web, per dispositivi mobili, IoT e di terze parti.
- *Back-end per dispositivi mobili:* è possibile creare back-end utilizzando Lambda e Amazon API Gateway per autenticare ed elaborare le richieste API.

Figura 2.32: Esempio di utilizzo di una funzione Lambda con Amazon S3

Concetti fondamentali di AWS Lambda

Per comprendere al meglio AWS Lambda è necessario introdurre i seguenti concetti fondamentali:

- *Funzione:* è una risorsa che è possibile invocare per eseguire il codice in Lambda. Una funzione dispone di codice per elaborare gli eventi trasmessi dall'utente o inviati da altri servizi AWS.
- *Trigger:* è una risorsa o una configurazione che richiama una funzione Lambda.
- *Evento:* è un documento in formato JSON formattato che contiene i dati che una funzione Lambda deve elaborare. Il runtime converte l'evento in un oggetto e lo passa al codice della funzione.
- *Ambiente di esecuzione:* gestisce le risorse necessarie per eseguire la funzione fornendo un ambiente di runtime sicuro e isolato. Quando si configura una funzione Lambda, è necessario specificare diversi parametri di configurazione, tra cui la quantità di memoria disponibile e il tempo massimo di esecuzione consentito per la funzione. Queste informazioni sono fondamentali perché Lambda possa stabilire l'ambiente di esecuzione in base alle esigenze specifiche della funzione. Una volta configurata, la funzione Lambda ed eventuali estensioni esterne, come runtime e librerie aggiuntive, vengono eseguite all'interno di un ambiente di esecuzione dedicato. In questo contesto, le autorizzazioni, le risorse, le credenziali e le variabili di ambiente sono condivise tra la funzione e le estensioni, garantendo coerenza e accesso uniforme alle risorse necessarie.

- *Architettura del set di istruzioni:* determina il tipo di processore del computer utilizzato da Lambda per eseguire la funzione. Lambda offre una scelta di architetture del set di istruzioni; questi sono:
  - *arm64:* architettura ARM a 64 bit, per il processore AWS Graviton2.
  - *x86_64:* architettura x86 a 64 bit, per processori basati su x86.
- *Pacchetti di implementazione:* utili per distribuire il codice della funzione Lambda, il quale supporta due tipi di pacchetti di implementazione: immagini di container[4] e archivi di file `.zip`.
- *Runtime:* fornisce un ambiente specifico del linguaggio di programmazione che viene eseguito in un ambiente di esecuzione. Il runtime inoltra eventi di chiamata, informazioni di contesto e risposte tra Lambda e la funzione.
- *Livello:* è un archivio di file, con estensione `.zip`, che può contenere codice, dati aggiuntivi, librerie, un runtime personalizzato, dati o file di configurazione. L'uso dei livelli riduce le dimensioni degli archivi di implementazione caricati e accelera la distribuzione del codice; inoltre, i livelli promuovono la condivisione del codice e la separazione delle responsabilità in modo da poter iterare più velocemente la scrittura della logica di business.
- *Estensione:* è una funzionalità che consente di integrare software di terze parti all'interno dell'ambiente di esecuzione delle funzioni Lambda. Le estensioni consentono agli sviluppatori di aggiungere funzionalità personalizzate, librerie, strumenti di monitoraggio, strumenti di sicurezza, e altro ancora, alle proprie funzioni Lambda senza dover modificare il codice della funzione stessa.
- *Simultaneità:* si riferisce al numero di richieste che la funzione è in grado di elaborare contemporaneamente in un determinato momento. Quando una funzione Lambda viene attivata da un evento, essa effettua il provisioning di un'istanza per gestire l'elaborazione dell'evento. Una volta che il codice della funzione Lambda ha completato l'esecuzione, l'istanza può, quindi, gestire ulteriori richieste. Se la funzione viene invocata nuovamente mentre un'altra richiesta è ancora in fase di elaborazione, Lambda effettua il provisioning di un'altra istanza, aumentando, così, la simultaneità della funzione.
- *Qualificatore:* quando si richiama o si visualizza una funzione, è possibile includere un qualificatore per specificare una versione o un alias. Una versione è uno snapshot immutabile del codice e della configurazione di una funzione.
- *Destinazione:* è una funzionalità che consente di indirizzare in modo automatico o manuale i risultati di un'operazione o un evento a una funzione Lambda specifica all'interno dell'ecosistema AWS.

2.4.8 Amazon DynamoDB

Amazon DynamoDB è un servizio di database NoSQL (Not Only SQL) completamente gestito che offre prestazioni elevate insieme a una scalabilità ottimale. DynamoDB permette di alleggerire il compito di gestione e scalare un database distribuito senza preoccuparsi del

[4] Un'immagine di container include il sistema operativo di base, il runtime, le estensioni Lambda, il codice dell'applicazione e le relative dipendenze. È, inoltre, possibile aggiungere all'immagine dati statici, come i modelli di machine learning.

provisioning dell'hardware, dell'installazione e della configurazione del software, della replica dei dati e del dimensionamento del cluster. Inoltre, esso offre funzionalità di crittografia dei dati inattivi, semplificando la protezione dei dati sensibili e riducendo la complessità operativa. DynamoDB offre la possibilità di creare tabelle di database in grado di archiviare e recuperare qualsiasi quantità di dati e di soddisfare qualsiasi livello di traffico di richieste. È possibile aumentare o ridurre la capacità di velocità effettiva delle tabelle senza tempi di inattività o diminuzione delle prestazioni. In aggiunta, DynamoDB fornisce anche funzionalità di backup on-demand, consentendo di creare backup completi delle tabelle per la conservazione e l'archiviazione a lungo termine; inoltre, è possibile abilitare il ripristino point-in-time che aiuta a proteggere le tabelle da operazioni di scrittura o cancellazione accidentali. Infine, esso consente di eliminare automaticamente gli elementi scaduti dalle tabelle per ridurre l'utilizzo dello spazio di archiviazione e il costo di archiviazione dei dati obsoleti.

Concetti e componenti principali di DynamoDB

I concetti e le componenti principali di DynamoDB sono i seguenti:

- *Tabelle, elementi e attributi:* in DynamoDB i dati sono organizzati all'interno di tabelle, analogamente a quanto avviene per altri DBMS. Ogni tabella contiene zero o più elementi, che sono insiemi di attributi univocamente identificabili tra loro. Ogni elemento è composto da uno o più attributi, che rappresentano elementi dati fondamentali e non richiedono ulteriori suddivisioni.

  La Figura 2.33 mostra una tabella denominata "Prenotazioni" con alcuni item e attributi di esempio.

ID *(Numero)*	cognome	giorno_prenotazione	nome	ora_prenotazione
823	mandorlini	2023-11-10	mattia	08:15
599	rossi	2023-10-26	giovanni	03:00
56	mandi	2023-10-23	mattia	17:00

Figura 2.33: Esempio di una tabella con DynamoDB

- *Chiave primaria:* identifica univocamente ciascun elemento della tabella, garantendo che nessun elemento possa condividere la stessa chiave. DynamoDB supporta due tipi di chiavi primarie:
  - *Chiave di partizione:* si tratta di una chiave primaria semplice, composta da un singolo attributo. DynamoDB utilizza il valore della chiave di partizione come input per una funzione hash interna, determinando, così, la partizione in cui verrà memorizzato l'elemento. In una tabella con una sola chiave di partizione, non è possibile che due elementi abbiano lo stesso valore su questo attributo.
  - *Chiave di partizione e chiave di ordinamento:* questo tipo di chiave primaria, noto come chiave primaria composta, è costituita da due attributi ovvero la chiave di partizione e la chiave di ordinamento. DynamoDB utilizza il valore della chiave di partizione per determinare la partizione in cui verrà memorizzato l'elemento e ordina gli elementi all'interno di questa partizione in base al valore della chiave di ordinamento. Mentre è possibile che più elementi abbiano lo stesso valore della chiave di partizione, devono avere diversi valori di chiave di ordinamento.

  Nell'esempio di Figura 2.33 è presente soltanto la chiave di partizione che è rappresentata dall'elemento ID.

- *Indici secondari:* consentono di eseguire query sui dati nella tabella utilizzando una chiave alternativa, oltre alle query sulla chiave primaria. DynamoDB non richiede l'utilizzo di indici, ma offre maggiore flessibilità alle applicazioni durante l'esecuzione di query sui dati. Esso supporta due tipi di indici secondari, ovvero:
  - *Indice secondario globale*, ossia un indice con una chiave di partizione e una chiave di ordinamento che possono essere differenti da quelle presenti sulla tabella.
  - *Indice secondario locale*, ossia un indice con la stessa chiave di partizione della tabella ma con una chiave di ordinamento diversa.
- *DynamoDB Streams:* è una funzionalità opzionale che cattura gli eventi di modifica dei dati nelle tabelle. I dati relativi a questi eventi vengono visualizzati nel flusso pressoché in tempo reale e nell'ordine in cui si sono verificati gli eventi stessi. Ogni evento è rappresentato da un record di flusso; se su una tabella viene abilitato un flusso, DynamoDB Streams scrive su questo record ogni volta che si verifica uno dei seguenti eventi:
  - *Una nuova voce viene aggiunta alla tabella*; il flusso acquisisce un'immagine dell'intera voce, inclusi tutti i suoi attributi.
  - *Una voce viene aggiornata*; il flusso acquisisce l'immagine "prima" e "dopo" della modifica di qualsiasi attributo nella voce.
  - *Una voce viene eliminata dalla tabella*; il flusso acquisisce un'immagine dell'intera voce prima che venga cancellata.

  Per comprendere meglio tale funzionalità consideriamo un esempio (Figura 2.34). Supponiamo di avere la tabella `Customers` contenente le informazioni sui clienti di un'azienda, e supponiamo di voler inviare una e-mail di benvenuto a ogni nuovo cliente. È possibile abilitare un flusso su questa tabella, e quindi associarlo ad una funzione Lambda. Tale funzione viene eseguita ogni volta che viene visualizzato un nuovo record di flusso, ma vengono elaborati solo i nuovi elementi aggiunti alla tabella `Customers` che hanno un attributo `EmailAddress`; a questo punto la funzione Lambda richiama Amazon Simple Email Service (Amazon SES) per inviare un messaggio e-mail a quell'indirizzo.

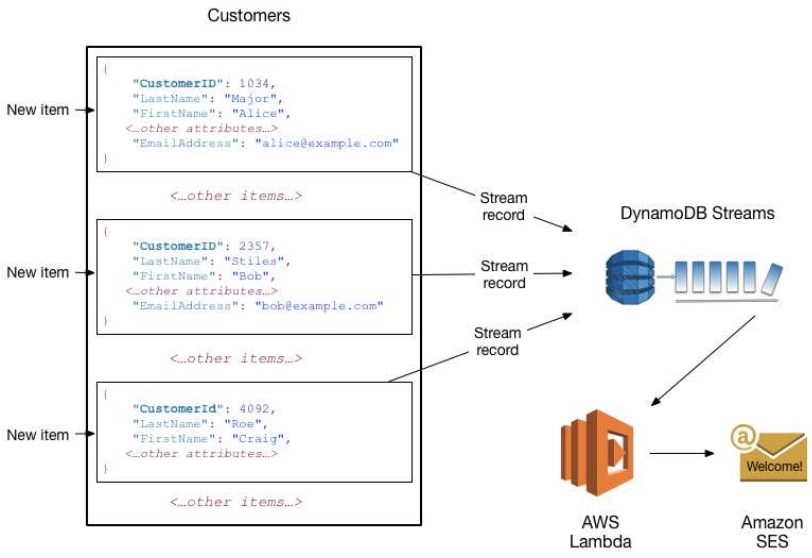


Figura 2.34: Esempio di funzionamento di DynamoDB Streams

2.4.9 Amazon CloudWatch

Amazon CloudWatch è un servizio di monitoraggio e osservabilità che consente di raccogliere, monitorare e gestire metriche, log e avvisi provenienti da altri servizi AWS o da applicazioni che fanno uso di AWS in tempo reale. In altre parole, con CloudWatch è possibile ottenere una visibilità a livello di sistema sull'utilizzo delle risorse, sulle prestazioni delle applicazioni e sullo stato operativo.

Concetti base di Amazon CloudWatch

La terminologia e i concetti seguenti sono fondamentali per la comprensione e l'uso di Amazon CloudWatch:

- *Spazio dei nomi:* un namespace funge da contenitore per le metriche correlate e offre un'organizzazione strutturata per i dati di monitoraggio. La sua importanza risiede nell'isolamento tra loro delle metriche, evitando l'aggregazione erronea di dati provenienti da diverse applicazioni o servizi nello stesso insieme di statistiche. Ciò assicura un'analisi precisa e mirata delle metriche, garantendo che le informazioni raccolte siano rilevanti per il contesto specifico di ciascuna applicazione o servizio monitorato.

- *Metriche:* rappresentano dati ordinati nel tempo che riflettono le prestazioni dei servizi AWS. Ogni metrica è come una variabile monitorata, mentre i punti dati rappresentano i valori che questa variabile assume nel tempo. Ogni punto dati deve essere associato a un timestamp, che fornisce un contesto temporale per l'analisi. Se non viene fornito un timestamp, CloudWatch ne crea automaticamente uno basato sull'ora di ricezione del punto dati.

- *Dimensioni:* sono attributi aggiuntivi associati alle metriche che consentono di organizzare, filtrare e analizzare i dati di monitoraggio in modo più dettagliato e contestuale, fornendo informazioni più approfondite sulle risorse monitorate.

- *Risoluzione:* rappresenta l'intervallo di tempo tra i punti dati di una metrica che determina la frequenza con cui CloudWatch raccoglie e registra i dati relativi alla metrica stessa.

- *Statistiche:* rappresentano aggregazioni di dati metrici raccolti durante periodi di tempo specifici che forniscono una visione chiara delle prestazioni dei servizi e delle risorse monitorate.

- *Percentili:* permettono di analizzare la distribuzione dei dati di una metrica e di comprendere le prestazioni dei servizi AWS. Calcolati su un periodo definito, aiutano a capire la distribuzione dei dati e a identificare tendenze e anomalie. Ad esempio, il 90° percentile indica il valore al di sotto del quale si trova il 90% dei dati osservati. CloudWatch utilizza l'interpolazione per calcolare i percentili, stimando valori quando i dati esatti non sono disponibili. I percentili sono accessibili tramite la console o l'API di CloudWatch e possono essere utilizzati per creare grafici e dashboard con l'obiettivo di monitorare le prestazioni dei servizi AWS.

- *Allarmi:* consentono di monitorare un singolo parametro durante un periodo definito e di eseguire azioni in base al valore del parametro stesso rispetto a una soglia specificata nel corso del tempo. Tali azioni possono comprendere l'invio di notifiche o l'applicazione di una policy di Auto Scaling, a seconda del superamento o del raggiungimento di determinati valori del parametro.

Funzionamento di Amazon CloudWatch

CloudWatch raccoglie metriche in tempo reale da una vasta gamma di servizi, tali metriche includono dati cruciali come l'utilizzo della CPU, della memoria e il traffico di rete, offrendo una panoramica completa delle prestazioni dei servizi. Una volta raccolte, le metriche vengono archiviate in CloudWatch, garantendo un'archiviazione a lungo termine che consente agli utenti di analizzare le prestazioni nel tempo e di individuare trend o anomalie. Gli utenti possono monitorare le metriche utilizzando la console di gestione AWS o l'API di CloudWatch, consentendo loro di creare grafici e dashboard personalizzati per monitorare le prestazioni dei servizi AWS in tempo reale. Inoltre, è possibile permettere agli utenti di impostare allarmi basati su metriche specifiche. Questi allarmi vengono attivati quando una metrica supera o scende al di sotto di una soglia predefinita, permettendo agli utenti di essere rapidamente avvisati di eventuali problemi o anomalie.
Un altro aspetto di CloudWatch è CloudWatch Logs che consente, di raccogliere, monitorare e gestire i log generati dai servizi e dalle applicazioni AWS. Gli utenti possono esplorare i log, filtrarli e creare metriche personalizzate basate sui dati contenuti in essi. Infine, CloudWatch si integra con altri servizi AWS, consentendo di utilizzare le metriche e i log per automatizzare azioni, scalare risorse, migliorare la sicurezza e molto altro ancora (Figura 2.35).

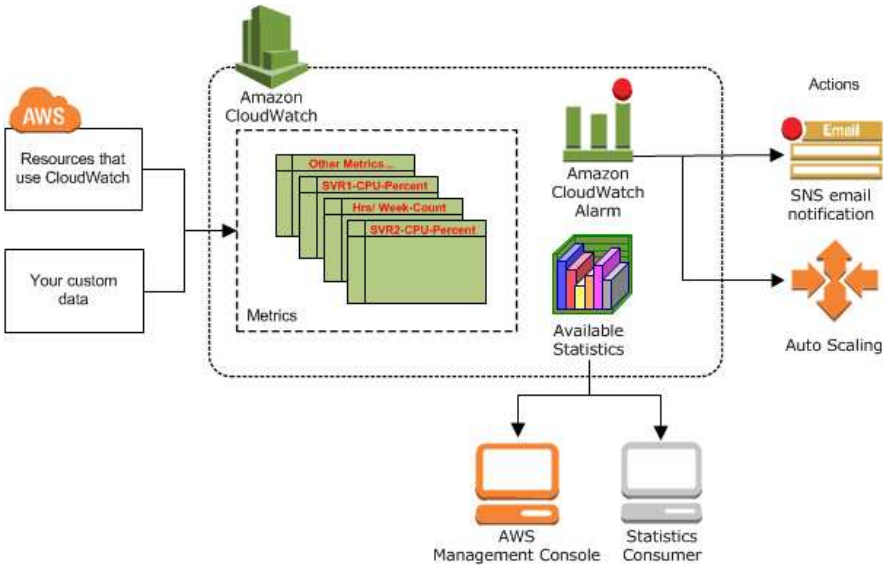


Figura 2.35: Schema di funzionamento di Amazon CloudWatch

CAPITOLO 3

Esperienza sui chatbot

In questo capitolo esamineremo approfonditamente una delle applicazioni dell'Intelligenza Artificiale: i chatbot. Cominceremo fornendo una definizione chiara di questa tecnologia e tracceremo un breve excursus storico. Successivamente, esploreremo gli utilizzi comuni e i vantaggi derivanti dall'impiego di tali strumenti e, infine, getteremo uno sguardo verso il futuro dei chatbot. Dopo aver completato questa fase introduttiva, ci addentreremo nell'analisi del case study, sviluppato utilizzando i servizi offerti da AWS. Approfondiremo le modalità di implementazione del chatbot e la sua distribuzione su una piattaforma per il testing.

3.1 Cos'è un chatbot

Il termine "Chatbot" deriva da "Chatterbot", coniato da Michael Mauldin nel 1994 per descrivere tutti i software conversazionali. Un chatbot è un programma software progettato per interagire con gli utenti attraverso una conversazione testuale o vocale. Questi sistemi utilizzano l'Intelligenza Artificiale (IA) e l'elaborazione del linguaggio naturale per comprendere le richieste degli utenti, elaborare le informazioni e fornire risposte adeguate o eseguire azioni specifiche. I primi chatbot erano essenzialmente programmi interattivi di domande frequenti (FAQ), programmati per rispondere a un insieme limitato di domande comuni con risposte predefinite. Incapaci di interpretare il linguaggio naturale, richiedevano generalmente agli utenti di selezionare semplici parole chiave e frasi per far proseguire la conversazione. Tali chatbot non erano in grado di elaborare domande complesse, né di rispondere a domande semplici non previste dagli sviluppatori.

Col passare del tempo, gli algoritmi dei chatbot sono diventati capaci di programmazioni basate su regole più complesse e ,persino, di elaborazione del linguaggio naturale, consentendo alle richieste dei clienti di essere espresse in modo conversazionale. I moderni chatbot utilizzano ora la comprensione del linguaggio naturale (NLU) per discernere il significato dell'input dell'utente, superando errori di battitura e problemi di traduzione. Gli strumenti IA avanzati mappano, quindi, quel significato all'intento specifico che l'utente vuole che il chatbot attui e utilizzano l'IA conversazionale per formulare una risposta appropriata. Questa sofisticazione, basata sui recenti progressi nei modelli di linguaggio di grandi dimensioni (LLM), ha portato a un aumento della soddisfazione dei clienti e ad applicazioni di chatbot più versatili.

Possiamo individuare due tipi principali di chatbot:

- *Chatbot dedicati alle attività (dichiarativi):* sono programmi dedicati a eseguire una specifica funzione e si concentrano sull'erogazione di risposte automatizzate, ma colloquiali,

alle richieste degli utenti. Utilizzano principalmente regole, NLP (Natural Language Processing) e una piccola quantità di machine learning. Le interazioni con questi chatbot sono altamente strutturate e specifiche, ad esempio per fornire assistenza o servizi, come risposte a domande frequenti o gestione di transazioni semplici. Sebbene utilizzino l'NLP per offrire un'esperienza di utilizzo intuitiva, le loro capacità rimangono di base. Attualmente, sono i chatbot più diffusi e utilizzati.

- *Chatbot predittivi basati sui dati (di conversazione):* gli assistenti virtuali, noti anche come assistenti digitali, sono più sofisticati dei chatbot dedicati alle attività e offrono un'interazione più partecipativa e personalizzata. Questi sistemi sono consapevoli del contesto circostante e utilizzano la comprensione del linguaggio naturale (NLU), l'elaborazione del linguaggio naturale (NLP) e il machine learning per apprendere e migliorare nel tempo. Gli assistenti virtuali applicano l'intelligenza predittiva e l'analisi dei dati per personalizzare l'esperienza degli utenti in base ai loro profili e comportamenti passati. Questi chatbot Possono imparare le preferenze degli utenti nel tempo, offrire raccomandazioni e anticipare le loro esigenze. Gli assistenti digitali possono anche avviare conversazioni e sono orientati al consumatore; esempi di questo tipo sono Siri di Apple e Alexa di Amazon.

3.2 Storia ed evoluzione dei chatbot

Fin dai primi stadi delle macchine e degli esperimenti tecnologici, si è sempre cercato di raggiungere un livello in cui i robot potessero agire e pensare in modo simile agli esseri umani. Pertanto, non sorprende che i primi esempi di chatbot risalgano al 1950. Con la sua mente visionaria, Alan Turing ideò un test rivoluzionario per valutare l'intelligenza delle macchine. Questo test sfidava i confini tra l'uomo e la macchina, richiedendo che un individuo non fosse in grado di distinguere se stesse interagendo con un software o con un essere umano.

Nel 1966 si verificò una delle più significative innovazioni nel campo degli chatbot. Joseph Weizenbaum, un rinomato matematico e crittografo, sviluppò un bot chiamato ELIZA, in grado di simulare conversazioni con uno psicoterapista. Le domande poste dal bot, basate sulle affermazioni dell'interlocutore, lo resero capace di superare brillantemente il test di Turing. Successivamente, nel 1972, Rollo Carpenter creò PARRY, uno dei primi chatbot in grado di dialogare con gli utenti, aprendo nuove prospettive nel campo della comunicazione uomo-macchina. Nel 1988, Kenneth Colby creò JABBERWACKY, un bot capace di simulare la conversazione di una persona affetta da schizofrenia paranoica, adottando una strategia di conversazione innovativa che lo rese più avanzato di ELIZA e descritto come un "ELIZA with attitude".

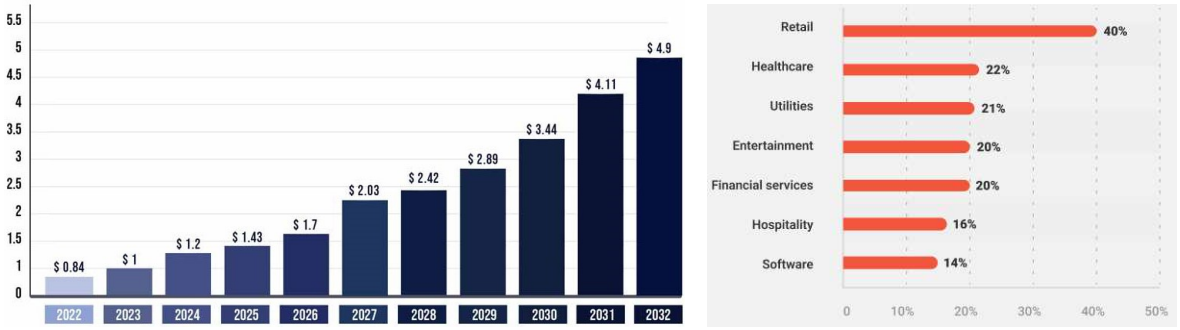
Negli anni '90, il Creative Lab contribuì al progresso sviluppando un programma di sintesi del linguaggio per i personal computer MS-DOS, dando origine alla competizione annuale sull'IA, che premiava i migliori bot capaci di simulare il comportamento umano basandosi sul test di Turing. Richard S. Wallace creò A.L.I.C.E. (Artificial Linguistic Internet Computer Entity), un bot capace di comprendere il linguaggio naturale e intrattenere conversazioni con gli utenti, rivelandosi un'autentica rivoluzione nel campo dell'IA, e vincendo il premio Loebner nel 2000, 2001 e 2004.

Nel 2001, Active Buddy, disponibile su piattaforme come MSN e AOL Instant Messenger, anticipò il futuro degli assistenti virtuali come Siri e S Voice, aprendo la strada a nuove forme di interazione con la tecnologia. Nello stesso anno, un gruppo di programmatori a San Pietroburgo diede vita a un bot chiamato Eugene Goostman, progettato per sembrare un tredicenne ucraino, il cui realismo fu tale che il 33% della giuria di un test di Turing credette di aver parlato con un umano.

Oggi, grazie alle tecniche avanzate di Natural Language Processing e Machine Learning, chatbot come IBM Watson, Siri, Google Assistant e Cortana stanno trasformando il modo in cui interagiamo con la tecnologia. Tuttavia, non tutto è stato sempre positivo, come dimostra l'esperienza con il bot Tay, che ha portato a esiti indesiderati e ha messo in luce le sfide e le responsabilità legate allo sviluppo di chatbot.

3.3 Usi comuni dei chatbot

L'impiego dei chatbot sta crescendo rapidamente in vari settori, portando soluzioni pratiche e innovative. Nel commercio elettronico, i chatbot semplificano operazioni come l'invio degli ordini e il pagamento, migliorando l'esperienza di acquisto online e personalizzando le raccomandazioni di prodotti. Settori come la medicina e il supporto legale utilizzano i chatbot per rispondere rapidamente alle domande più comuni degli utenti, semplificando l'accesso alle informazioni pertinenti. Inoltre, nell'ambito della scoperta di notizie e contenuti, i chatbot offrono risposte personalizzate alle preferenze degli utenti. Questo rivoluziona l'interazione con i servizi online, garantendo un supporto sempre disponibile. Inoltre, i chatbot ottimizzano l'esperienza di gestione dei servizi IT, automatizzando processi come la gestione delle password e il monitoraggio del sistema. A livello aziendale, essi sono utilizzati nei contact center per indirizzare le comunicazioni in entrata e semplificare le attività interne come l'onboarding dei dipendenti. Per i consumatori, i chatbot forniscono servizi come l'acquisto di biglietti e la comparazione di prodotti, estendendosi anche a settori bancari, retail e alimentari. Nel settore pubblico, i chatbot gestiscono richieste di servizi urbani e risolvono problemi di fatturazione, offrendo soluzioni efficienti e accessibili. Nella Figura 3.1 sono presentati i risultati di due studi: uno riguardante la futura evoluzione del mercato dei chatbot e l'altro focalizzato sul loro utilizzo comune nelle aziende.



(a) Mercato globale dei chatbot (USD Billion) **(b)** Casi d'uso dei chatbot nelle aziende

Figura 3.1: Analisi del mercato e degli usi dei chatbot

3.4 Rischi e futuro dei chatbot

Ogni vantaggio offerto da un chatbot può trasformarsi in uno svantaggio se si utilizzano la piattaforma, la programmazione o i dati sbagliati. I tradizionali chatbot di Intelligenza Artificiale sono in grado di fornire un servizio clienti rapido, ma presentano limitazioni in quanto molti si appoggiano ai sistemi basati su regole che automatizzano compiti e forniscono risposte predefinite alle richieste dei clienti. I nuovi chatbot di Intelligenza Artificiale generativa possono portare con sé rischi di sicurezza, tra cui la minaccia di perdita di dati, la confidenzialità non conforme agli standard, preoccupazioni sulla responsabilità, complessità

della proprietà intellettuale, licenze incomplete dei dati sorgente e incertezze sulla privacy e sul rispetto delle leggi internazionali. Inoltre, la mancanza di dati di input adeguati può comportare il rischio continuo di "allucinazioni", ovvero risposte inaccurate o non pertinenti che costringono il cliente a spostare la conversazione su un altro canale. La sicurezza e la perdita di dati rappresentano un rischio se informazioni sensibili di terze parti o interne all'azienda vengono inserite in un chatbot di Intelligenza Artificiale generativa, diventando parte del suo modello di dati che potrebbe essere condiviso con altri soggetti che pongono domande pertinenti. Ciò potrebbe portare a perdite di dati e a violare le politiche di sicurezza dell'organizzazione. Guardando sia all'attuale che alla futura evoluzione dei chatbot, si intravede un miglioramento progressivo che li renderà più affidabili nel tempo. Già dal 2024, si prevede che i modelli di Intelligenza Artificiale generativa diventeranno più sofisticati e versatili, consentendo un utilizzo più ampio in varie applicazioni. Questi avanzamenti non solo renderanno i chatbot più accoglienti e intelligenti, ma contribuiranno anche a ridurre la diffusione di informazioni erronee, rendendo le conversazioni con i chatbot più precise e affidabili. I chatbot rivoluzioneranno il servizio clienti tramite piattaforme automatizzate che garantiranno risposte immediate alle email e alle telefonate, ridefinendo le aspettative dei consumatori. Essi saranno in grado di estrarre informazioni da diverse fonti, diventando multimodali, mentre le capacità di integrazione avanzate permetteranno una collaborazione personalizzata con altri sistemi. Le imprese adotteranno modelli di IA personalizzati per migliorare la qualità delle interazioni e l'efficacia delle applicazioni, comportando, secondo un rapporto di Grand View Research, una crescita annuale del mercato globale dell'IA del 37,7% tra il 2023 e il 2030. Emergeranno assistenti IA che semplificheranno i compiti umani, migliorando sia i processi aziendali che le performance individuali. Tuttavia, con l'avvento dell'IA generativa, si è aperta una discussione sui diritti d'autore, poiché questa forma di IA consente ai computer di creare contenuti simili a quelli umani, sollevando interrogativi su chi abbia diritto di utilizzare o distribuire tali contenuti. Questa discussione potrebbe portare a una revisione delle leggi sul copyright per adattarle ai tempi che cambiano.

3.5 Descrizione del case study

In questa sezione tratteremo nel dettaglio il primo case study sviluppato che consiste nella realizzazione di un chatbot mediante il servizio Amazon Lex di AWS.

3.5.1 Obiettivo dell'esperienza

L'idea fondamentale si basa sull'implementazione di un chatbot che fornisca assistenza e servizi utili ai membri di una palestra e/o di un centro fitness, con l'obiettivo finale di integrarlo su una piattaforma di messaggistica, come Telegram, per il testing. I servizi si suddividono in due categorie principali che, a loro volta, sono suddivise in altri servizi specifici. In particolare abbiamo:

- *Servizi legati all'assistenza clienti:* questi servizi agevolano l'accesso alla struttura e permettono ai membri di conoscere tutte le informazioni a loro utili; per esempio, è possibile:
  - Effettuare, visualizzare e cancellare una prenotazione.
  - Ottenere informazioni come la posizione della struttura, gli orari di apertura, nonché i prezzi, ed eventuali sconti, sugli abbonamenti.
- *Servizi legati alla consulenza fitness:* questo gruppo di servizi fornisce consigli, informazioni e strumenti utili in ambito fitness, in particolare su quattro aspetti:

- *Allenamento:* l'utente, sulla base dei suoi obiettivi e di alcune domande precedentemente fornite, riceverà una serie di consigli per ottimizzare le sessioni di allenamento.
- *Alimentazione:* con questa funzionalità l'utente potrà conoscere il suo dispendio calorico giornaliero (TDEE), grazie alla conoscenza di dati personali precedentemente forniti al chatbot, e potrà, di conseguenza, calibrare la sua dieta in maniera più opportuna per il suo obiettivo.
- *Riposo e Recupero:* questa funzionalità fornisce consigli per ottimizzare il riposo e il recupero dopo una sessione di allenamento sulla base delle risposte dell'utente.
- *Monitoraggio dei progressi:* quest'ultimo servizio permette all'utente di registrare e visualizzare le sue sessioni di allenamento e i pasti effettuati durante la giornata.

Per la realizzazione sono stati utilizzati, oltre ad Amazon Lex, altri servizi accessori che hanno contribuito a sviluppare tutte le funzionalità precedentemente citate, come:

- *Amazon DynamoDB*, per il salvataggio e la gestione dei dati, come le prenotazioni, gli allenamenti e i pasti.
- *AWS Lambda*, per poter elaborare messaggi di risposta e dati, come il calcolo del TDEE e l'aggiunta, la rimozione e la visualizzazione di informazioni inerenti alle prenotazioni, ai pasti e agli allenamenti.
- *Amazon API Gateway*, per la realizzazione di un'API che permettesse la comunicazione tra Telegram e il chatbot di Amazon Lex per il testing.

3.5.2 Analisi degli intent

Passeremo, ora, ad analizzare tutti gli intent definiti, descrivendo il loro scopo, la struttura, gli slot utilizzati e le modalità di adempimento alla richiesta dell'utente.

Benvenuto

Nella Figura 3.2 vengono mostrate la struttura dell'intent e le possibili alternative.

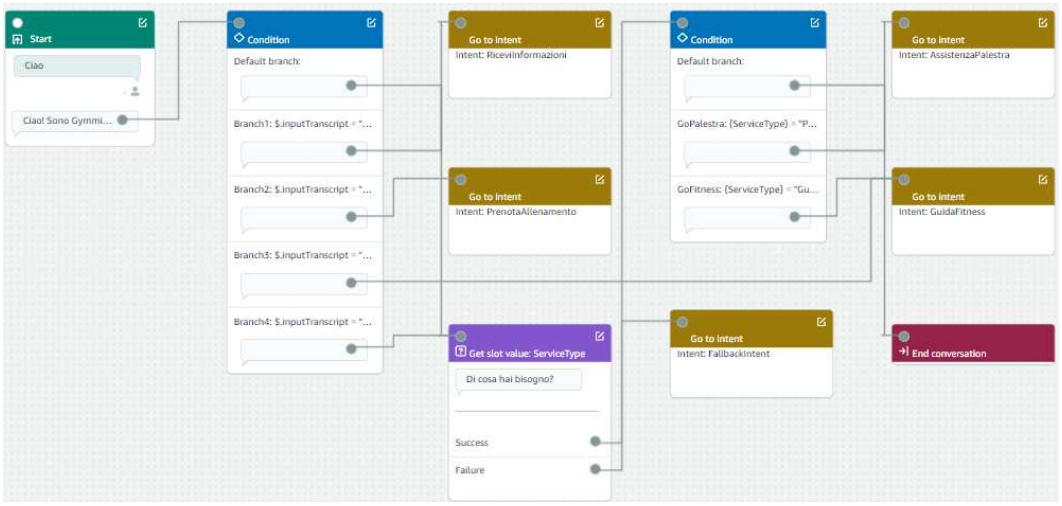


Figura 3.2: Struttura dell'intent di benvenuto

Questo intent ha l'obiettivo di presentare all'utente le possibili funzionalità che il chatbot offre; inoltre, esso è stato anche progettato per poter accedere direttamente a tali servizi

senza ripetere la procedura di presentazione. Come per ogni intent, è necessario fornire alcuni esempi di frasi e/o parole che l'utente potrebbe scrivere per attivarlo; in questo caso, l'utente può utilizzare qualsiasi saluto iniziale, oppure parole specifiche, per richiedere un determinato servizio (Figura 3.3).

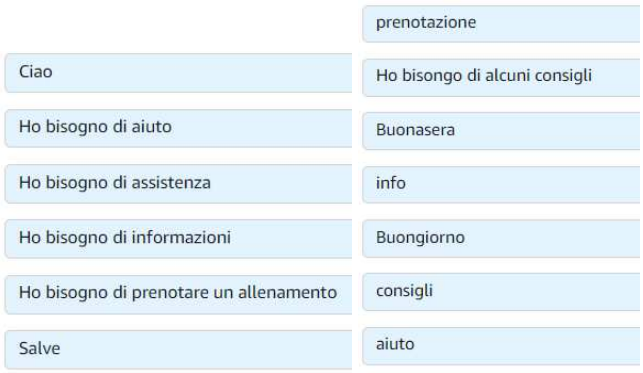


Figura 3.3: Affermazioni di esempio per l'intent di benvenuto

In seguito, esaminando la struttura dell'intent, si può notare che il flusso inizia non appena l'utente fornisce una delle dichiarazioni di esempio illustrate nella figura precedente. Il flusso procede, quindi, in un blocco "condition" in cui viene valutata la richiesta dell'utente. In base a questa valutazione, è possibile attivare automaticamente un altro intent oppure attivare lo slot[1] (Figura 3.4). Una volta che l'utente ha risposto, il chatbot attiva l'intent corrispondente al servizio richiesto, concludendo, così, il flusso.

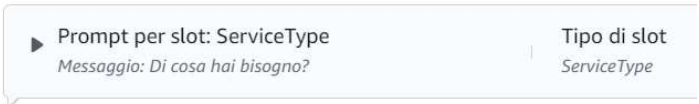


Figura 3.4: Slot per la selezione del servizio desiderato

Nella Figura 3.5 sono riportate le risposte del chatbot alle richieste dell'utente.

Figura 3.5: Test dell'intent di benveuto

[1]Lo slot è implementato utilizzando un tipo personalizzato (ServiceType) che accetta solo risposte simili a quelle definite durante la creazione.

AssistenzaPalestra

Nella Figura 3.6 vengono mostrate la struttura dell'intent e le possibili alternative.

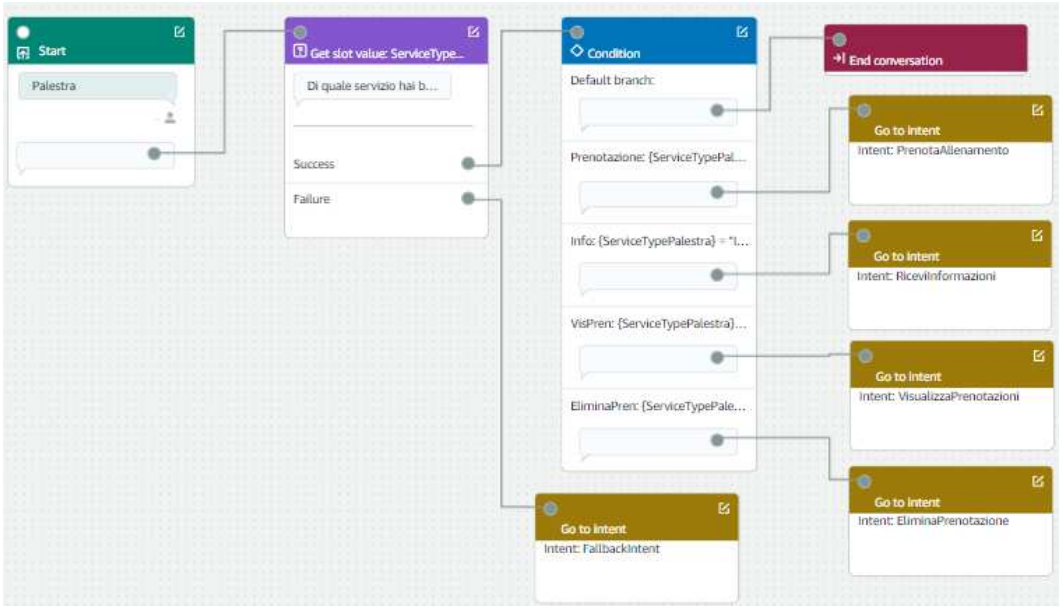


Figura 3.6: Struttura dell'intent per l'assistenza clienti

Quando l'utente digita "Palestra" oppure fa intendere che desidera assistenza, tale intent si attiva, mostra i servizi disponibili, e con l'uso di uno slot personalizzato chiede all'utente cosa necessità. Il blocco "condition" valuta la risposta e attiva di conseguenza un altro intent per soddisfarla. Di seguito, nelle Figure 3.7 e 3.8, sono riportati, rispettivamente, lo slot utilizzato e i risultati del test.

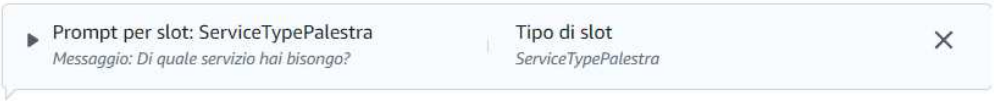


Figura 3.7: Slot per la selezione del servizio di assistenza desiderato

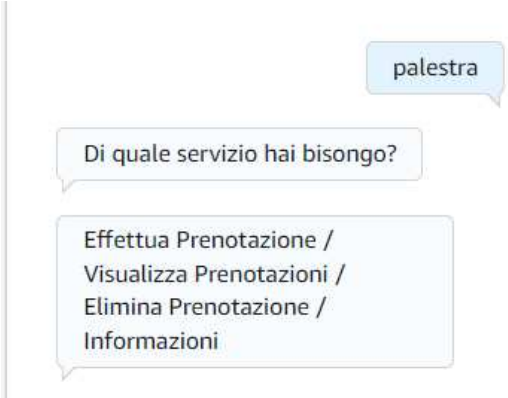


Figura 3.8: Test dell'intent AssistenzaPalestra

RiceviInformazioni

Nella Figura 3.9 vengono mostrate la struttura dell'intent e le possibili alternative.

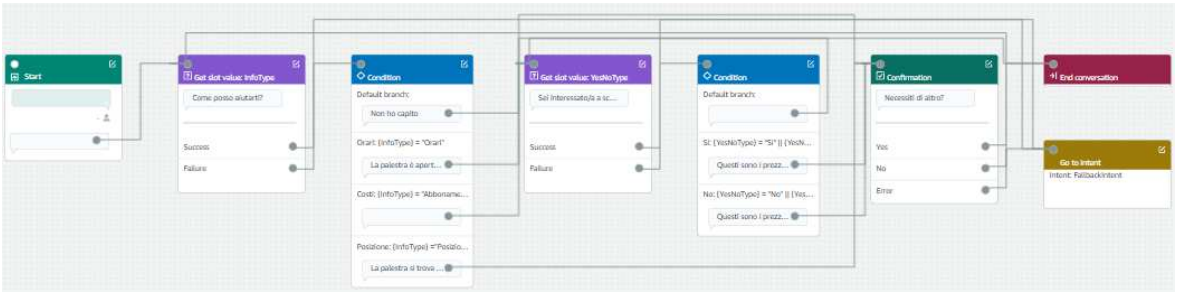


Figura 3.9: Struttura dell'intent RiceviInformazioni

Questo intent, tramite uno slot, chiede all'utente di selezionare quale informazione desidera ricevere tra orari, posizione e prezzi degli abbonamenti. Una volta ottenuta una risposta, il chatbot provvede a fornire l'informazione richiesta e in seguito, usando uno slot di conferma, chiede all'utente se necessita di altro. In caso di risposta affermativa, il chatbot reitera l'intent, altrimenti lo conclude. Nelle Figure 3.10 e 3.11 vengono mostrati gli slot usati e un esempio di esecuzione dell'intent.

Figura 3.10: Slot per la selezione delle informazioni necessarie e per la conferma

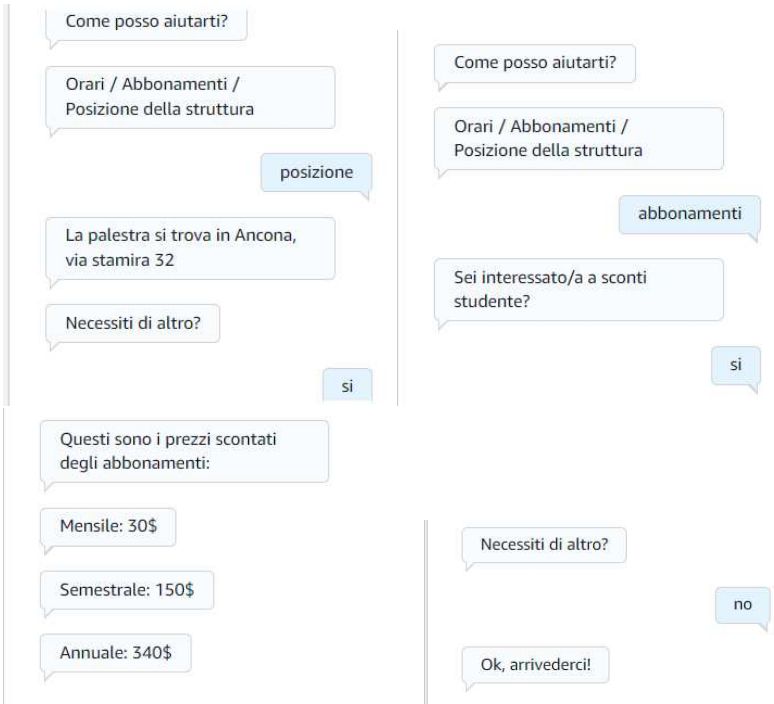


Figura 3.11: Test dell'intent RiceviInformazioni

PrenotaAllenamento

Nella Figura 3.12 vengono mostrate la struttura dell'intent e le possibili alternative.

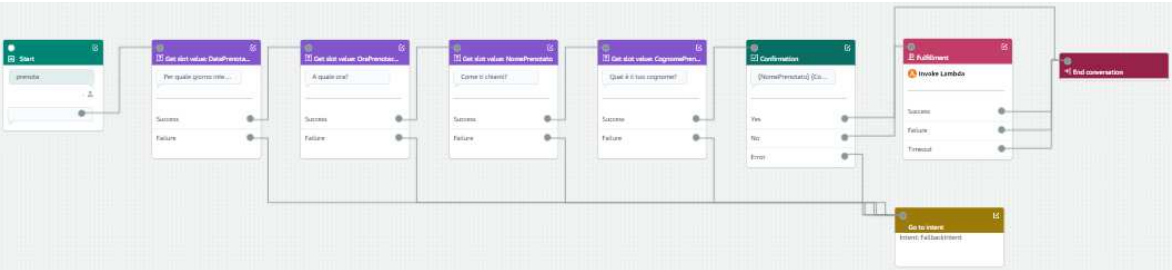


Figura 3.12: Struttara dell'intent per la prenotazione in palestra

Nella Figura 3.13 viene illustrata, sotto varie forme, la possibilità con cui l'utente può far attivare questo intent[2]. In seguito alla richiesta, il chatbot procede a chiedere alcune informazioni all'utente per adempiere in maniera corretta all'esigenza (Figura 3.14) e, soltanto alla fine, chiede di confermare l'operazione o, di annullarla.

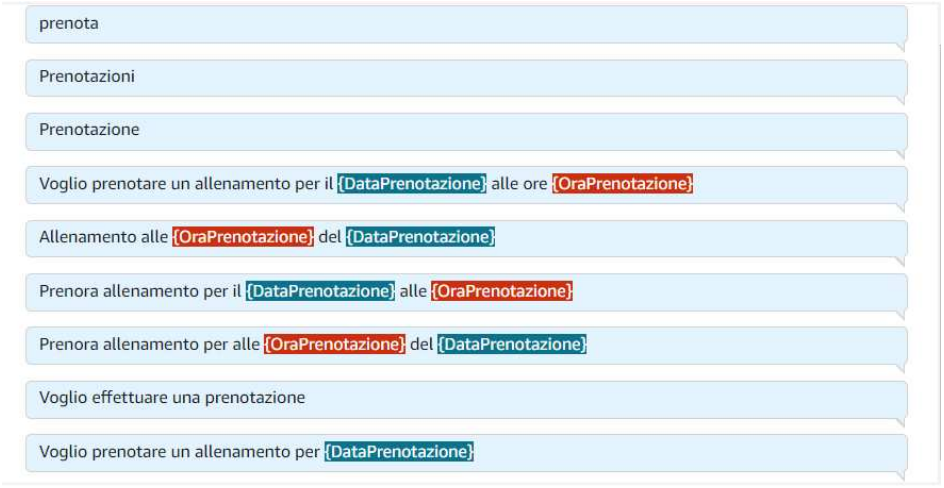


Figura 3.13: Frasi di esempio per l'intent di prenotazione

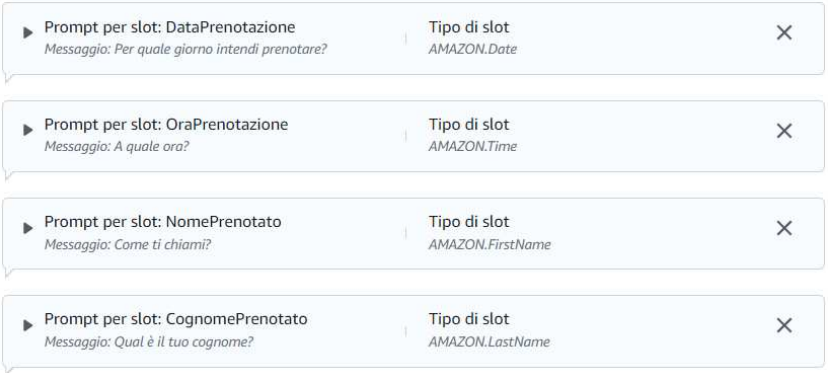


Figura 3.14: Slot dell'intent per la prenotazione

[2]Il formato particolare che hanno alcune frasi di esempio è utile per fornire il valore dello slot nel momento in cui l'utente, già nella richiesta, specifica le informazioni di cui ha bisogno il chatbot, evitando, così, di chiederle in seguito.

Un particolare che è possibile notare dalla struttura dell'intent è che vi è un blocco di adempimento chiamato "Invoke Lambda". Questo blocco permette di assegnare il compito di gestire la richiesta dell'utente ad una funzione Lambda che avrà il compito di prelevare i dati della prenotazione e di salvarli sul database. Di seguito, nella Figura 3.15, è riportato il test dell'intent.

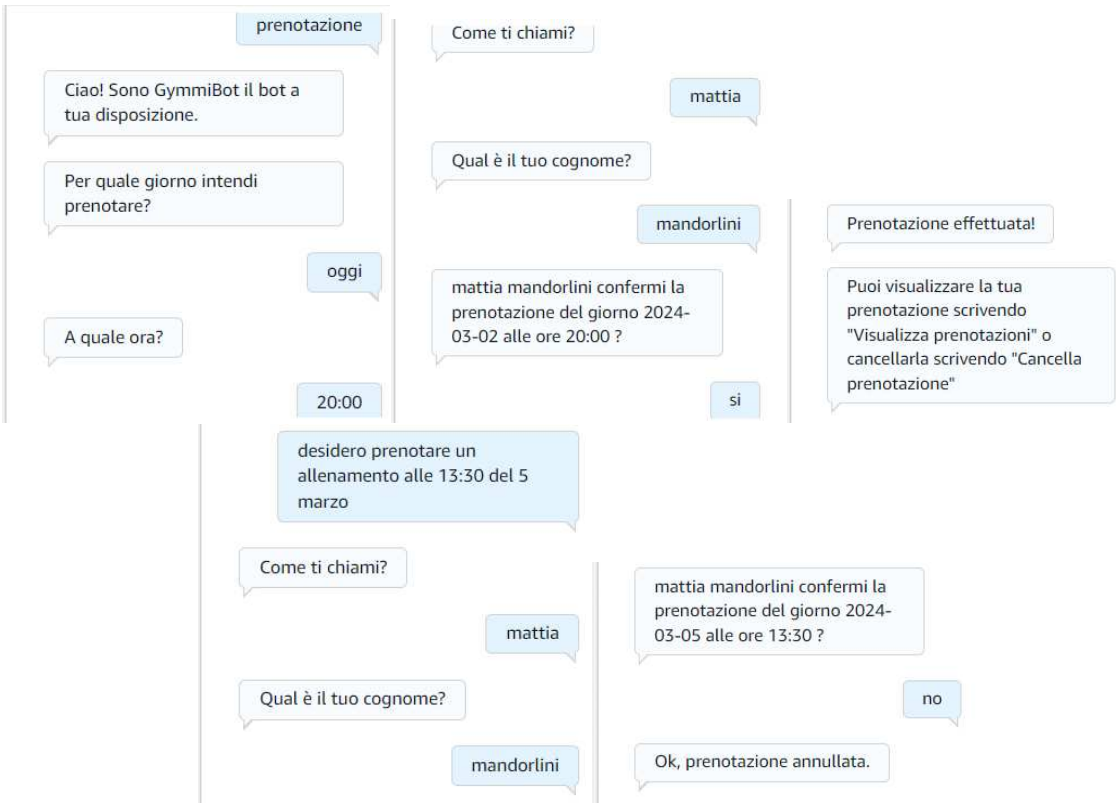


Figura 3.15: Test dell'intent per la prenotazione

VisualizzaPrenotazioni

Nella Figura 3.16 vengono mostrate la struttura dell'intent e le possibili alternative.

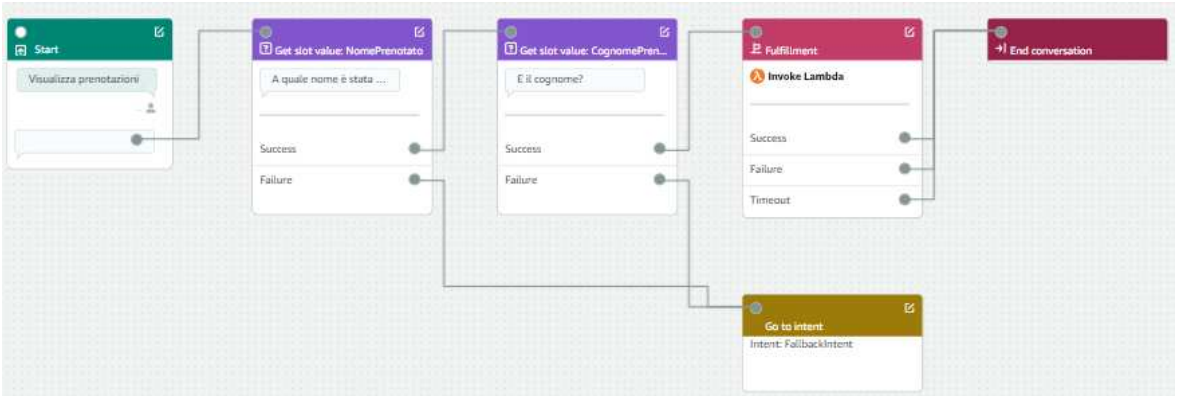


Figura 3.16: Struttura dell'intent VisualizzaPrenotazioni

Questo intent permette di mostrare tutte le prenotazioni effettuate da un utente. L'intent viene attivato utilizzando frasi di esempio, come quelle mostrate in Figura 3.17, e procede

chiedendo informazioni, come nome e cognome, per poter ricercare nel database, tramite la funzione lambda associata, i dati di interesse. Nella Figura 3.18 vengono evidenziati gli slot utilizzati mentre nella Figura 3.19 vengono mostrati i risultati delle esecuzioni dell'intent.

Figura 3.17: Frasi di esempio dell'intent VisualizzaPrenotazioni

Figura 3.18: Slot dell'intent VisualizzaPrenotazioni

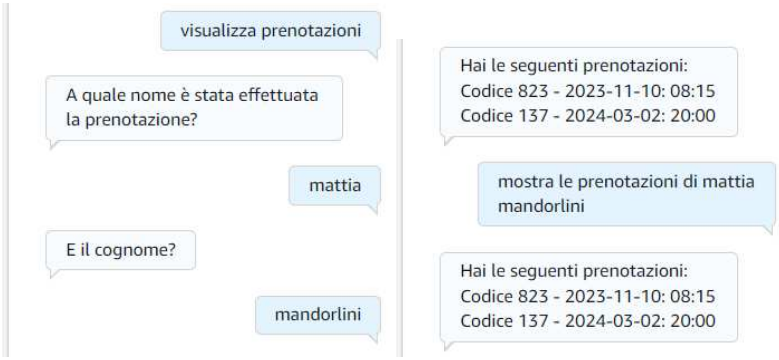


Figura 3.19: Test dell'intent VisualizzaPrenotazioni

EliminaPrenotazioni

Nella Figura 3.20 vengono mostrate la struttura dell'intent e le possibili alternative.

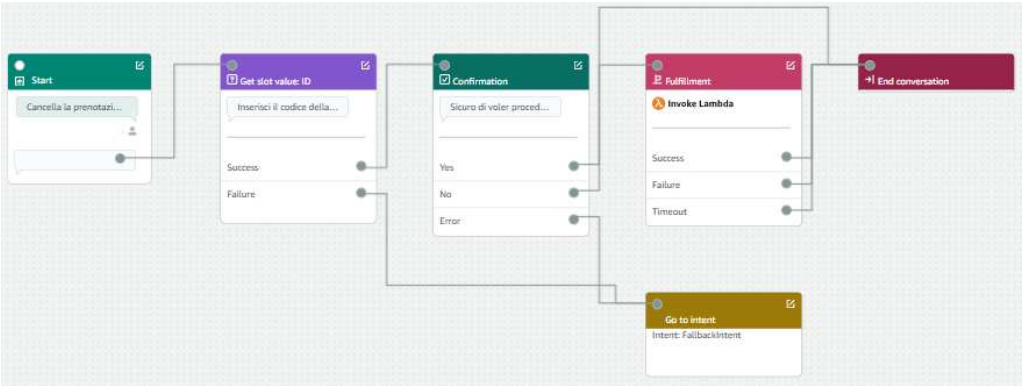


Figura 3.20: Struttura dell'intent EliminaPrenotazioni

Tale intent permette di cancellare una prenotazione utilizzando il codice (Figura 3.21) ad essa associato; tale codice viene richiesto dal chatbot tramite uno slot (Figura 3.22) oppure viene fornito direttamente dall'utente nel caso in cui egli ne sia già a conoscenza. Il codice viene elaborato dalla funzione Lambda al momento del salvataggio sul database e viene, poi, riutilizzato per effettuare la cancellazione.

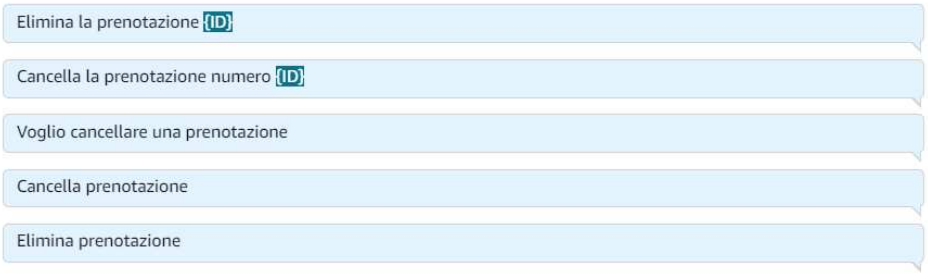


Figura 3.21: Frasi di esempio dell'intent EliminaPrenotazioni

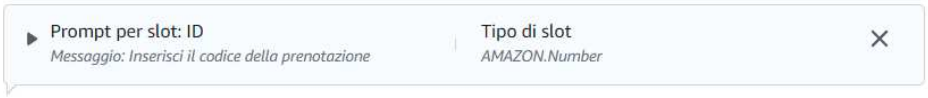


Figura 3.22: Slot dell'intent EliminaPrenotazioni

Successivamente, nella Figura 3.23, è possibile osservare il funzionamento dell'intent.

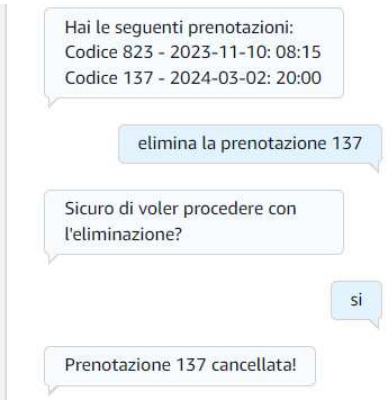


Figura 3.23: Test dell'intent EliminaPrenotazioni

GuidaFitness

Nella Figura 3.24 vengono mostrate la struttura dell'intent e le possibili alternative. La sua funzionalità è simile a quella dell'intent AssistenzaPalestra, ovvero, esso deve introdurre all'utente la seconda categoria di funzionalità offerte dal chatbot, quelle inerenti alla consulenza fitness. Per fare ciò, vengono richiesti nome e cognome e vengono presentati i vari servizi disponibili tramite degli slot (Figura 3.25) che serviranno per gestire i vari utenti e tener traccia delle loro attività. Per quanto riguarda questi ultimi aspetti, una funziona Lambda avrà il compito di elaborare e salvare i dati per fornire risposte adeguate. Di seguito, nella Figura 3.26, viene mostrato un esempio di funzionamento.

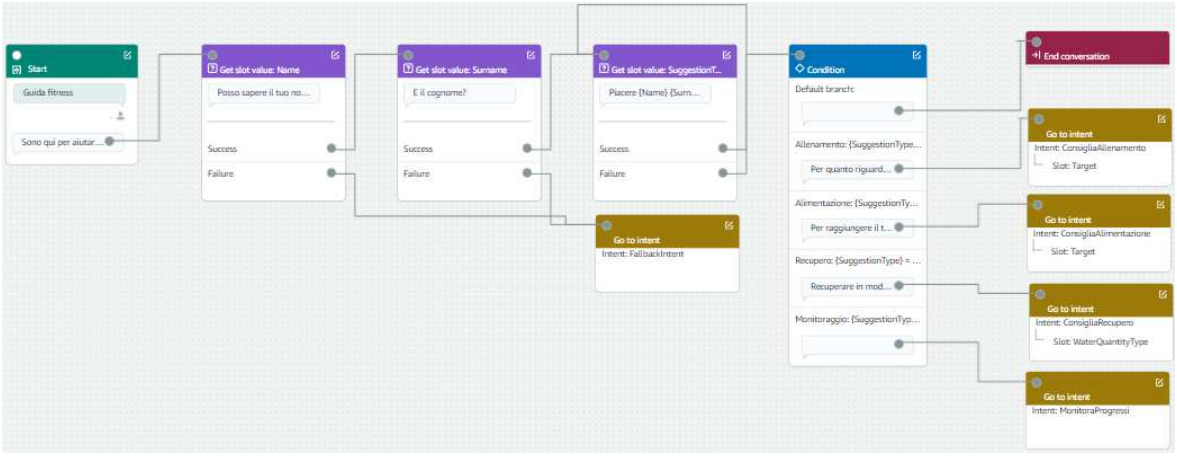


Figura 3.24: Struttura dell'intent GuidaFitness

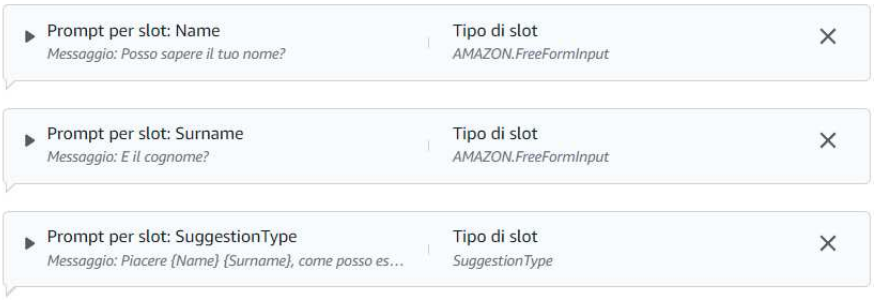


Figura 3.25: Slot dell'intent GuidaFitness

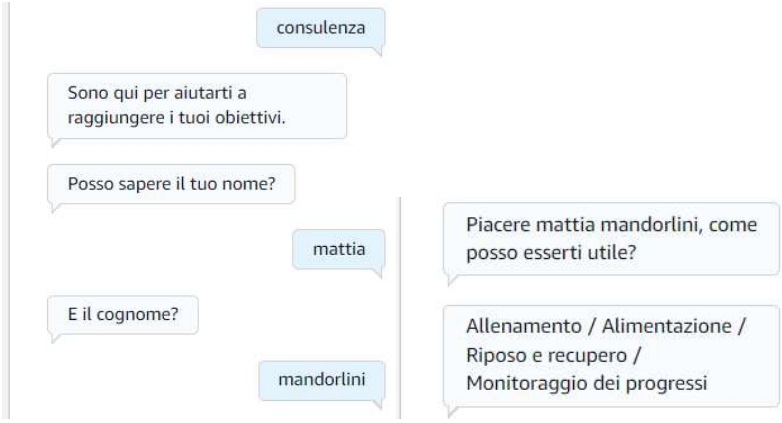


Figura 3.26: Test dell'intent GuidaFitness

ConsigliaAllenamento

Nella Figura 3.27 vengono mostrate la struttura dell'intent e le possibili alternative.

Quando l'utente digita "Allenamento" come risposta all'intent GuidaFitness, il chatbot attiverà questo intent. Il suo scopo è quello di fornire consigli su come strutturare il proprio allenamento in base all'obiettivo, il tempo a disposizione e il luogo in cui ci si allena (Figura 3.28). Pertanto, le risposte fornite dal chatbot saranno diverse a seconda delle possibilità dell'utente. Nella Figura 3.29 è riportato un esempio specifico di funzionamento con un utente che ha l'obiettivo di dimagrire allenandosi in palestra con una o due ore a disposizione.

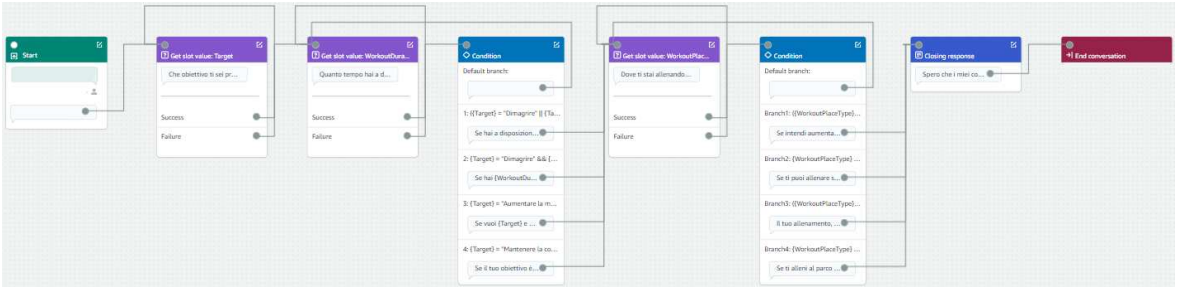


Figura 3.27: Struttura dell'intent ConsigliaAllenameto

Figura 3.28: Slot dell'intent ConsigliaAllenamento

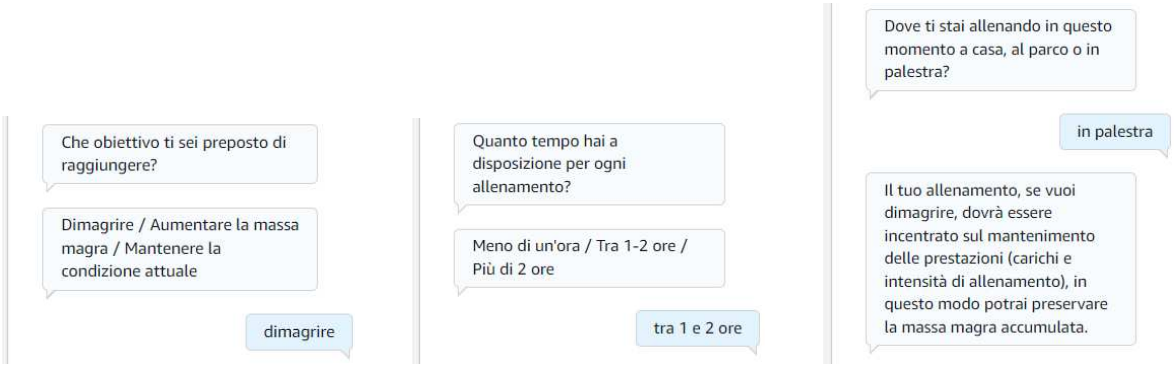


Figura 3.29: Test dell'intent ConsigliaAllenamento

ConsigliaAlimentazione

Nella Figura 3.30 vengono mostrate la struttura dell'intent e le possibili alternative. A seguito della richiesta "Alimentazione" come risposta all'intent GuidaFitness, si attiverà questa funzionalità con lo scopo di fornire all'utente una stima del suo fabbisogno calorico giornaliero (TDEE), con l'obiettivo di avere un riferimento per la gestione della propria alimentazione. Questa procedura avviene richiedendo varie informazioni personali tramite degli slot (Figura 3.31); una volta raccolte tali informazioni, le stesse verranno elaborate da una funzione Lambda per il calcolo del risultato finale. In Figura 3.32 è riportato un esempio di applicazione.

ConsigliaRecupero

Nella Figura 3.33 vengono mostrate la struttura dell'intent e le possibili alternative. Digitando "Riposo e recupero", il chatbot inizierà a fornire consigli generali; successivamente, sulla base delle risposte fornite dall'utente ad alcune domande (Figura 3.34), approfondirà

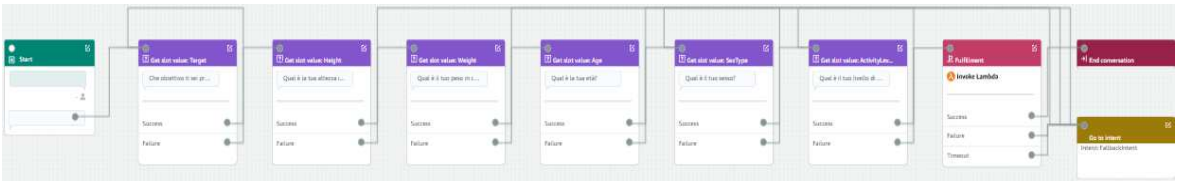


Figura 3.30: Struttura dell'intent ConsigliaAlimentazione

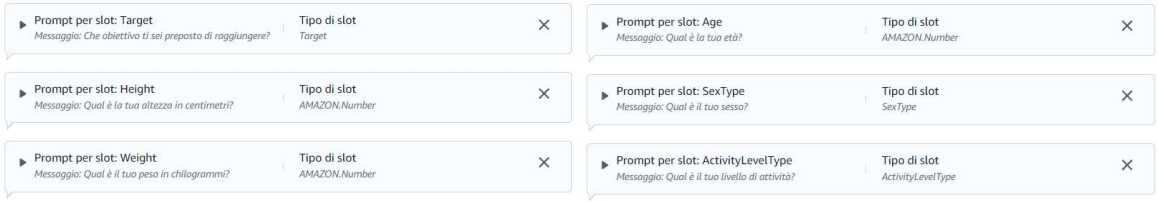


Figura 3.31: Slot dell'intent ConsigliaAlimentazione

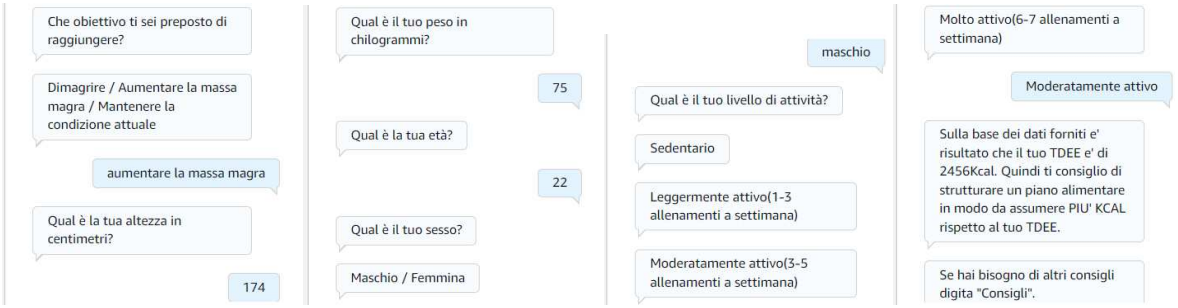


Figura 3.32: Test dell'intent ConsigliaAlimentazione

alcuni aspetti per permettergli di migliorare in quei particolari punti. Nella Figura 3.35 viene mostrato un esempio di funzionamento.

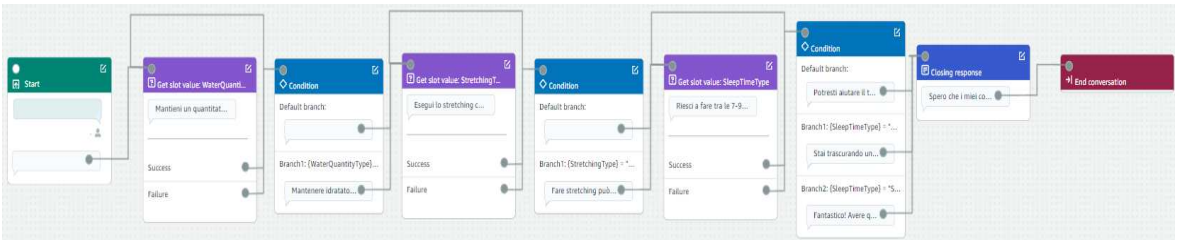


Figura 3.33: Struttura dell'intent ConsigliaRecupero

Figura 3.34: Slot dell'intent ConsigliaRecupero

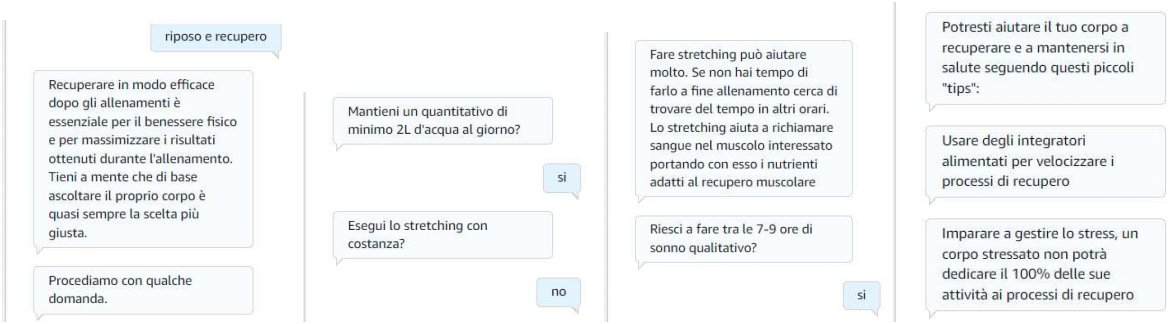


Figura 3.35: Test dell'intent ConsigliaRecupero

MonitoraProgressi

Nella Figura 3.36 vengono mostrate la struttura dell'intent e le possibili alternative. Quest'ultimo servizio di consulenza fitness si dirama in quattro sotto-servizi. Ciascuno di essi viene presentato grazie allo slot utilizzato (Figura 3.37), permettendo all'utente di selezionare il servizio che gli interessa. Il funzionamento è illustrato nella Figura 3.38.

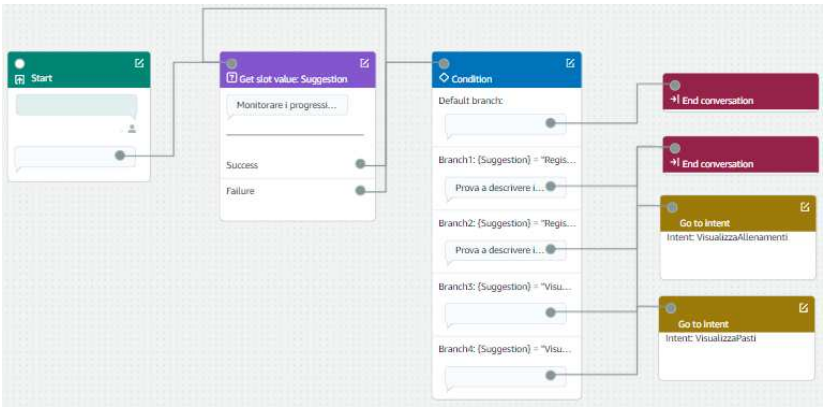


Figura 3.36: Struttura dell'intent MonitoraProgressi

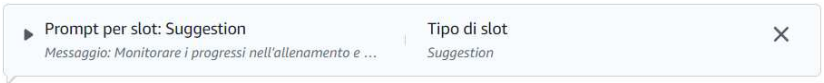


Figura 3.37: Slot dell'intent MonitoraProgressi

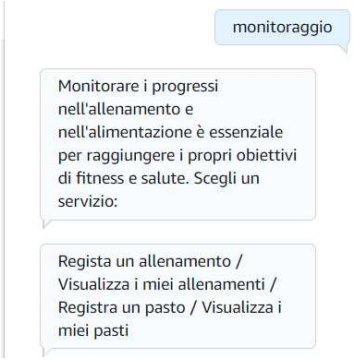


Figura 3.38: Test dell'intent MonitoraProgressi

RegistraPasto

Nella Figura 3.39 vengono mostrate la struttura dell'intent e le possibili alternative.

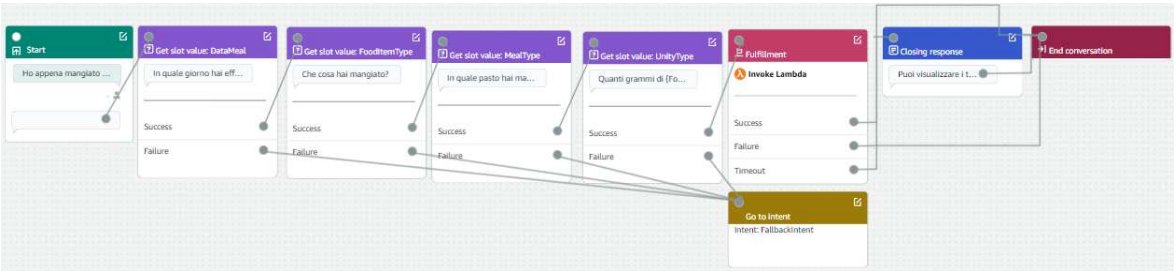


Figura 3.39: Struttura dell'intent RegistraPasto

Con questa funzionalità l'utente può registrare i pasti effettuati nei diversi momenti della giornata in modo da avere una stima delle quantità di cibo consumate. L'intent si può svolgere in due modi in base alla volontà dell'utente; tramite la prima modalità il chatbot chiede, passo dopo passo, tutti i dati necessari per compiere la registrazione; la seconda modalità, invece, consiste nel far descrivere all'utente il pasto che ha effettuato includendo tutti i dati necessari, dopodiché si elabora la richiesta e si estrapolano tutte le informazioni per compilare gli slot (Figura 3.41). Tale procedura avviene fornendo varie frasi di esempio (Figura 3.40) in modo tale che il chatbot possa riconoscere vari modi di formulare la richiesta e adempiere al compito. Nelle Figure 3.42 e 3.43 sono mostrati i due modi per eseguire l'intent.

Figura 3.40: Frasi di esempio dell'intent RegistraPasto

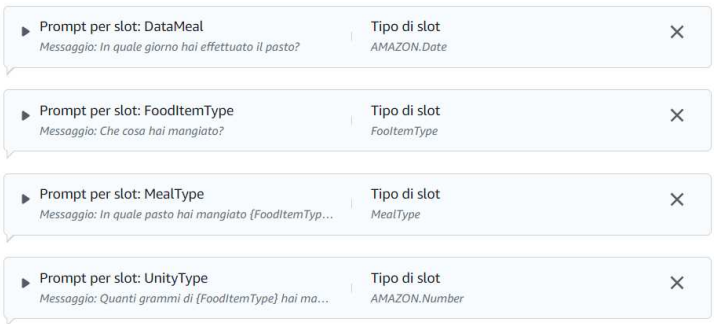


Figura 3.41: Slot dell'intent RegistraPasto

Figura 3.42: Prima modalità di test dell'intent RegistraPasto

Figura 3.43: Seconda modalità di test dell'intent RegistraPasto

VisualizzaPasti

Nella Figura 3.44 vengono mostrate la struttura dell'intent e le possibili alternative.

Figura 3.44: Struttura dell'intent VisualizzaPasti

Il suo scopo è quello di permettere all'utente di visualizzare i pasti da lui effettuati sulla base della data inserita. Lo slot utilizzato permette di raccogliere questa informazione che poi viene elaborata dalla funzione Lambda per ricavare i dati dal database (Figura 3.45). Di seguito è mostrato il funzionamento dell'intent (Figura 3.46).

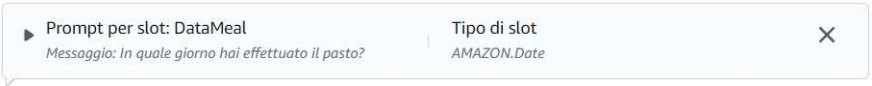


Figura 3.45: Slot dell'intent VisualizzaPasti

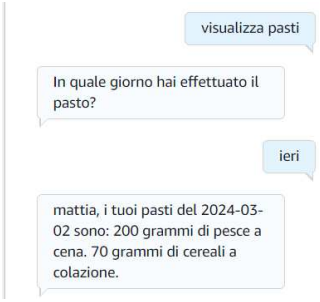


Figura 3.46: Test dell'intent VisualizzaPasti

RegistraAllenamento

Nella Figura 3.47 vengono mostrate la struttura dell'intent e le possibili alternative.

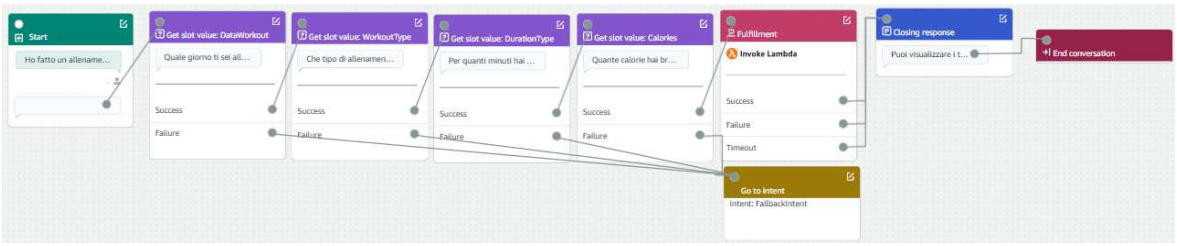


Figura 3.47: Struttura dell'intent RegistraAllenamento

Con questa funzionalità l'utente può registrare i suoi allenamenti in modo da tenere traccia delle kcal bruciate. L'intent può essere gestito in due modalità, a seconda della preferenza dell'utente. In una delle opzioni, il chatbot richiede passo dopo passo tutte le informazioni necessarie per completare la registrazione. In alternativa, invece, si invita l'utente a descrivere l'allenamento svolto. Successivamente, il sistema elabora la richiesta ed estrae le informazioni necessarie per compilare gli slot (Figura 3.49). Questo processo viene facilitato fornendo diverse frasi di esempio (Figura 3.48), in modo che il chatbot possa identificare varie formulazioni della richiesta e soddisfare il compito. Nelle Figure 3.50 e 3.51 sono mostrati i due modi per eseguire l'intent.

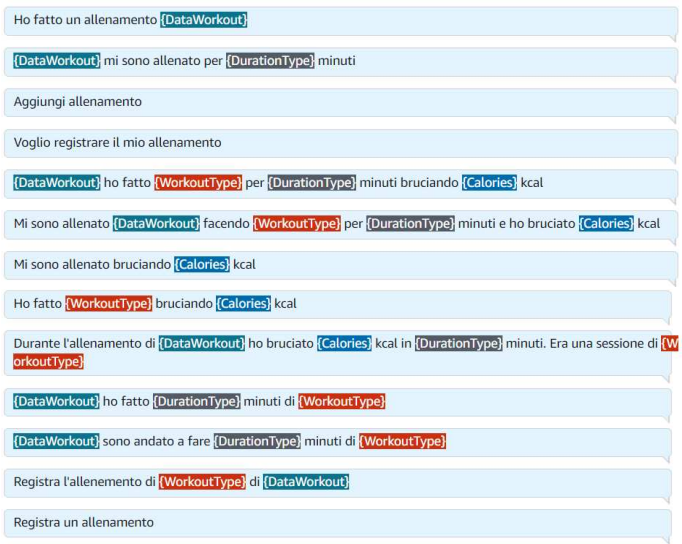


Figura 3.48: Frasi di esempio dell'intent RegistraAllenamento

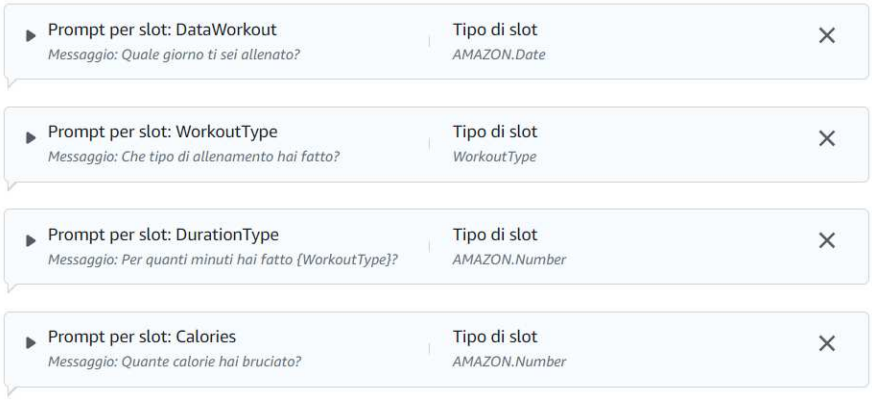


Figura 3.49: Slot dell'intent RegistraAllenamento

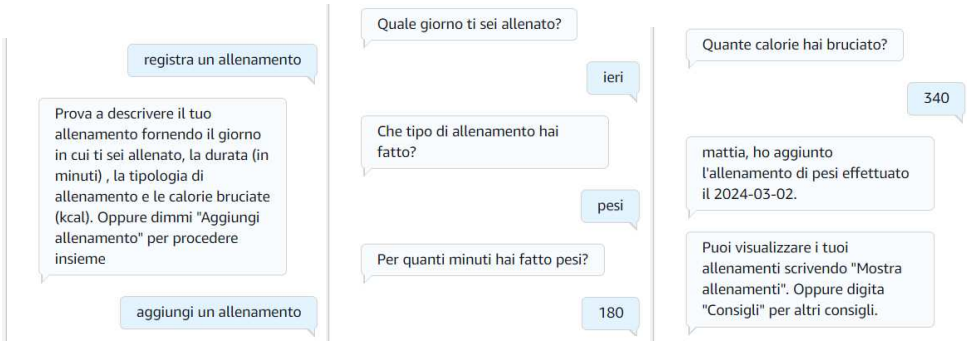


Figura 3.50: Prima modalità di test dell'intent RegistraAllenamento

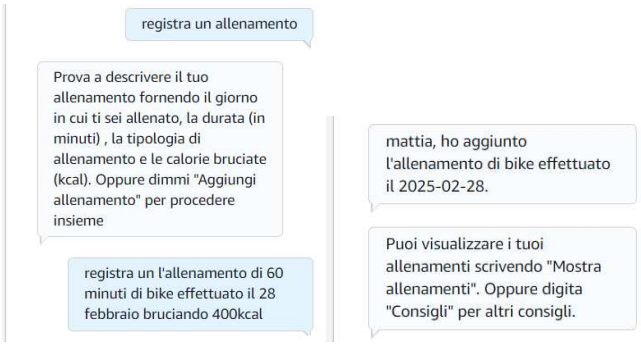


Figura 3.51: Seconda modalità di test dell'intent RegistraAllenamento

VisualizzaAllenamenti

Nella Figura 3.52 vengono mostrate la struttura dell'intent e le possibili alternative. Il suo scopo è quello di permettere all'utente di visualizzare i suoi allenamenti sulla base della data inserita. Lo slot utilizzato permette di raccogliere questa informazione che viene, poi, elaborata dalla funzione Lambda per ricavare i dati dal database (Figura 3.53). Di seguito è mostrato il funzionamento dell'intent (Figura 3.54).

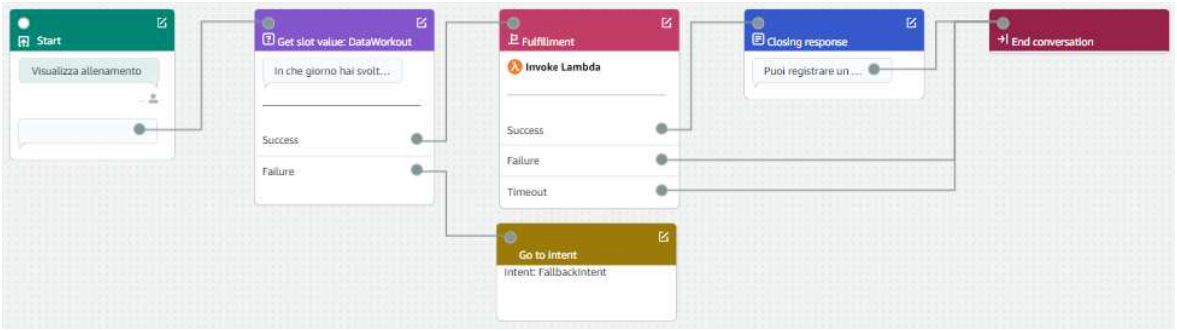


Figura 3.52: Struttura dell'intent VisualizzaAllenamenti

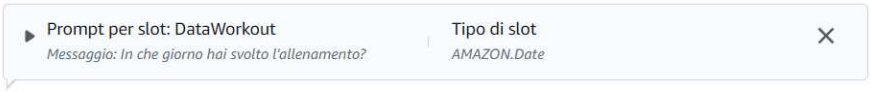


Figura 3.53: Slot dell'intent VisualizzaAllenamenti

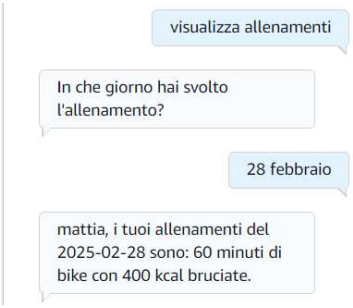


Figura 3.54: Test dell'intent VisualizzaAllenamenti

FallbackIntent

Il "fallback intent" in Amazon Lex è un'intento predefinito che si attiva quando il sistema non riesce a comprendere l'intenzione dell'utente dall'input fornito. Di solito, offre un messaggio standard oppure delle opzioni per aiutare l'utente a chiarire la richiesta o a essere indirizzato verso ulteriori supporti. Esso serve a migliorare l'esperienza dell'utente assicurando che, anche se il chatbot non interpreta correttamente l'input, l'utente riceva comunque una risposta utile per continuare la conversazione.

Di seguito nelle Figure 3.55, 3.56 e 3.57 sono illustrate le informazioni contenute nel database DynamoDB per quanto riguarda i pasti, gli allenamenti e le prenotazioni degli utenti.

ID *(Numero)*	alimento	cognome_utente	data_pasto	grammi	nome_utente	pasto
213	pesce	mandorlini	2024-03-02	200	mattia	cena
250	cereali	mandorlini	2024-03-02	70	mattia	colazione

Figura 3.55: Tabella DynamoDB relativa ai pasti registrati

ID (Numero)	allenamento	calorie	cognome_utente	data_allenamento	durata	nome_utente
194	cardio	300	mandorlini	2024-03-02	40	mattia
640	pesi	340	mandorlini	2024-03-02	180	mattia
716	bike	400	mandorlini	2025-02-28	60	mattia

Figura 3.56: Tabella DynamoDB relativa agli allenamenti registrati

ID (Numero)	cognome	giorno_prenotazione	nome	ora_prenotazione
56	mandi	2023-10-23	mattia	17:00
823	mandorlini	2023-11-10	mattia	08:15
599	rossi	2023-10-26	giovanni	03:00

Figura 3.57: Tabella DynamoDB relativa alle prenotazioni effettuate

3.5.3 Funzione Lambda BotHandler

In questa sottosezione osserveremo più nel dettaglio la funzione Lambda di gestione del chatbot. Come accennato durante l'analisi degli intent, tale funzione permette l'interazione con il database DynamoDB e la generazione di risposte sulla base dei dati in ingresso ed in uscita. Andando più nel dettaglio, la funzione è stata realizzata utilizzando il runtime `Python 3.11` e, come tipo di pacchetto di esportazione, il `.zip`. Maggiori dettagli sono mostrati nella Figura 3.58.

```
10 Resources:
11   BotHandler:
12     Type: AWS::Serverless::Function
13     Properties:
14       CodeUri: .
15       Description: ''
16       MemorySize: 128
17       Timeout: 3
18       Handler: lambda_function.lambda_handler
19       Runtime: python3.11
20       Architectures:
21         - x86_64
22       EphemeralStorage:
23         Size: 512
24       EventInvokeConfig:
25         MaximumEventAgeInSeconds: 21600
26         MaximumRetryAttempts: 2
27       PackageType: Zip
```

Figura 3.58: Modello AWS di configurazione della funzione `BotHandler`

Il BotHandler ha come entry point[3] la funzione `lambda_handler`, che riceve come argomenti un `event` e un `context`. Soltanto il primo è di fondamentale importanza, poiché contiene tutte le informazioni provenienti dal chatbot di Amazon Lex, come il nome dell'intent, i valori degli slot e altri dati di contorno. È proprio con l'uso di queste informazioni che è stata strutturata la funzione Lambda, che invoca altre sue sotto funzioni in base all'intent attivato dall'utente (Figura 3.59).

Una aspetto sul quale è importante soffermarsi è `boto3`, ovvero, una libreria sviluppata e progettata per semplificare l'interazione con AWS tramite codice Python, fornendo un'API

[3] L'entry point address è l'indirizzo in cui si troverà l'istruzione iniziale da eseguire dopo che il processo sarà stato caricato in memoria dal loader. In altre parole, è la prima istruzione che viene eseguita.

```python
import json
import boto3
import random

def prepare_response_close(event, msgText):
def prepare_response(event, msgText,msgText2):
def add_workout(event):
def show_workout(event):
def add_meal(event):
def show_meal(event):
def find_tdee(event):
def add_reservation(event):
def show_reservation(event):
def remove_reservation(event):
```

(a) Import delle librerie e definizione delle sotto funzioni

```python
def lambda_handler(event, context):
    intent_name = event['sessionState']['intent']['name']
    response = None

    if intent_name == 'RegistraAllenamento':
        response = add_workout(event)
    elif intent_name == "VisualizzaAllenamenti":
        response = show_workout(event)
    elif intent_name == "VisualizzaPasti":
        response = show_meal(event)
    elif intent_name == 'RegistraPasto':
        response = add_meal(event)
    elif intent_name == 'ConsigliaAlimentazione':
        response = find_tdee(event)
    elif intent_name == "PrenotaAllenamento":
        response = add_reservation(event)
    elif intent_name == "VisualizzaPrenotazioni":
        response = show_reservation(event)
    elif intent_name == "EliminaPrenotazione":
        response = remove_reservation(event)

    return response
```

(b) Corpo della funzione principale

Figura 3.59: Struttura della funzione Lambda `BotHandler`

semplice e intuitiva per accedere e utilizzare i servizi AWS. Diamo, ora, un'occhiata più approfondita a ciascuna delle sotto funzioni elencate nella figura precedente.

add_workout

Questa funzione preleva tutti i valori degli slot dell'intent RegistraAllenamento dal parametro `event` e, attraverso `boto3`, crea una connessione al servizio DynamoDB di AWS. Una volta fatto ciò, con l'API `put_item` è possibile salvare nella tabella specificata i dati desiderati. Alla fine viene ritornato al chatbot un messaggio di risposta da far visualizzare all'utente una volta che l'elaborazione è andata a buon fine (Figura 3.60).

```python
def add_workout(event):
    table_name = 'Allenamenti'

    data = event['sessionState']['intent']['slots']['DataWorkout']['value']['interpretedValue']
    allenamento = event['sessionState']['intent']['slots']['WorkoutType']['value']['interpretedValue']
    durata = event['sessionState']['intent']['slots']['DurationType']['value']['interpretedValue']
    calorie = event['sessionState']['intent']['slots']['Calories']['value']['interpretedValue']
    nome_utente = event['sessionState']['sessionAttributes']['nome_utente']
    cognome_utente = event['sessionState']['sessionAttributes']['cognome_utente']

    dynamodb = boto3.client('dynamodb')

    id = random.randint(1, 999)

    dynamodb.put_item(
        TableName=table_name,
        Item={
            "ID": {"N": str(id)},
            "data_allenamento": {"S": data},
            "allenamento": {"S": allenamento},
            "durata": {"N": str(durata)},
            "calorie":{"N": str(calorie)},
            "nome_utente": {'S': nome_utente},
            "cognome_utente": {'S': cognome_utente},
        },
    )
    msgText = f"{nome_utente}, ho aggiunto l'allenamento di {allenamento} effettuato il {data}."
    msgText2 = 'Puoi visualizzare i tuoi allenamenti scrivendo "Mostra allenamenti".\nOppure digita "Consigli" per altri consigli.'

    return prepare_response(event,msgText,msgText2)
```

Figura 3.60: Funzione `add_workout` di `BotHandler`

show_workout

Questa funzione utilizza il valore dello slot contenente la data dell'allenamento e il nome e cognome dell'utente per effettuare un query al database e prelevare i dati di tutti gli allenamenti effettuati da quell'utente. Tale operazione avviene tramite l'API `scan` di `boto3` che permette di filtrare e ricercare nella tabella specificata. Alla fine dell'elaborazione viene restitutito all'utente un messaggio di risposta (Figura 3.61).

```
def show_workout(event):
    table_name = 'Allenamenti'
    dynamodb = boto3.client('dynamodb')

    dataW = event['sessionState']['intent']['slots']['DataWorkout']['value']['interpretedValue']
    nome_utente = event['sessionState']['sessionAttributes']['nome_utente']
    cognome_utente = event['sessionState']['sessionAttributes']['cognome_utente']

    response = dynamodb.scan(
        TableName=table_name,
        FilterExpression='data_allenamento= :dataW and nome_utente = :nome_utente and cognome_utente = :cognome_utente',
        ExpressionAttributeValues={
            ':dataW': {'S': dataW},
            ':nome_utente': {'S': nome_utente},
            ':cognome_utente': {'S': cognome_utente},
        },
    )

    # Estrai gli elementi dalla risposta
    items = response.get('Items', [])
    response_text = ""

    if not items:
        return prepare_response_close(event, f"{nome_utente} non hai effettuato nessun allenamento il {dataW}.")

    response_text = f"{nome_utente}, i tuoi allenamenti del {dataW} sono:\n"
    for item in items:
        response_text += f"{item['durata']['N']} minuti di {item['allenamento']['S']} con {item['calorie']['N']} kcal bruciate."

    msgText= 'Puoi registrare i tuoi allenamenti scrivendo "Registra allenamento".\nOppure digita "Consigli" per altri consigli.'

    return prepare_response(event, response_text,msgText)
```

Figura 3.61: Funzione `show_workout` di `BotHandler`

add_meal

Le operazioni effettuate da questa funzione sono analoghe a quelle della funzione `add_workout` (Figura 3.62).

```
def add_meal(event):
    table_name = 'Pasti'

    mealtype = event['sessionState']['intent']['slots']['MealType']['value']['interpretedValue']
    food = event['sessionState']['intent']['slots']['FoodItemType']['value']['interpretedValue']
    unity = event['sessionState']['intent']['slots']['UnityType']['value']['interpretedValue']
    data = event['sessionState']['intent']['slots']['DataMeal']['value']['interpretedValue']
    nome_utente = event['sessionState']['sessionAttributes']['nome_utente']
    cognome_utente = event['sessionState']['sessionAttributes']['cognome_utente']

    dynamodb = boto3.client('dynamodb')

    id = random.randint(1, 999)

    dynamodb.put_item(
        TableName=table_name,
        Item={
            "ID": {"N": str(id)},
            "data_pasto": {"S": data},
            "alimento": {"S": food},
            "pasto": {"S": mealtype},
            "grammi": {"N": str(unity)},
            "nome_utente": {'S': nome_utente},
            "cognome_utente": {'S': cognome_utente},
        },
    )

    msgText = f"{nome_utente}, ho aggiunto il pasto effettuato il {data} a {mealtype}."
    msgText2 = 'Puoi visualizzare i tuoi pasti scrivendo "Mostra pasti".\nOppure digita "Consigli" per altri consigli.'

    return prepare_response(event,msgText,msgText2)
```

Figura 3.62: Funzione `add_meal` di `BotHandler`

show_meal

Le operazioni effettuate da questa funzione sono analoghe a quelle della funzione `show_workout` (Figura 3.63).

```
def show_meal(event):
    table_name = 'Pasti'
    dynamodb = boto3.client('dynamodb')

    dataM = event['sessionState']['intent']['slots']['DataMeal']['value']['interpretedValue']
    nome_utente = event['sessionState']['sessionAttributes']['nome_utente']
    cognome_utente = event['sessionState']['sessionAttributes']['cognome_utente']

    response = dynamodb.scan(
        TableName=table_name,
        FilterExpression='data_pasto= :dataM and nome_utente = :nome_utente and cognome_utente = :cognome_utente',
        ExpressionAttributeValues={
            ':dataM': {'S': dataM},
            ':nome_utente': {'S': nome_utente},
            ':cognome_utente': {'S': cognome_utente},
        },
    )

    # Estrai gli elementi dalla risposta
    items = response.get('Items', [])
    response_text = ""

    if not items:
        return prepare_response_close(event, f"{nome_utente} non hai effettuato nessun pasto il {dataM}.")

    response_text += f"{nome_utente}, i tuoi pasti del {dataM} sono:\n"
    for item in items:
        response_text += f"{item['grammi']['N']} grammi di {item['alimento']['S']} a {item['pasto']['S']}.\n"

    msgText = 'Puoi registrare i tuoi pasti scrivendo "Registra pasto".\nOppure digita "Consigli" per altri consigli.'

    return prepare_response(event,response_text,msgText)
```

Figura 3.63: Funzione `show_meal` di `BotHandler`

add_reservation

Le operazioni effettuate da questa funzione sono analoghe a quelle di `add_workout` e `add_meal` (Figura 3.64).

```
def add_reservation(event):
    dynamodb = boto3.client('dynamodb')
    table_name = 'Prenotazioni'

    day = event['sessionState']['intent']['slots']['DataPrenotazione']['value']['interpretedValue']
    time = event['sessionState']['intent']['slots']['OraPrenotazione']['value']['interpretedValue']
    name = event['sessionState']['intent']['slots']['NomePrenotato']['value']['interpretedValue'].lower()
    surname = event['sessionState']['intent']['slots']['CognomePrenotato']['value']['interpretedValue'].lower()

    id = random.randint(1, 999)

    dynamodb.put_item(
        TableName=table_name,
        Item={
            "ID": {"N": str(id)},
            "giorno_prenotazione": {"S": day},
            "ora_prenotazione": {"S": time},
            "nome": {"S": name},
            "cognome": {"S": surname},
        },
    )

    msgText = "Prenotazione effettuata!"
    msgText2 = 'Puoi visualizzare la tua prenotazione scrivendo "Visualizza prenotazioni" o cancellarla scrivendo "Cancella prenotazione"'

    return prepare_response(event,msgText,msgText2)
```

Figura 3.64: Funzione `add_reservation` di `BotHandler`

show_reservation

Le operazioni effettuate da questa funzione sono analoghe a quelle di `show_workout` e `show_meal` (Figura 3.65).

```python
def show_reservation(event):
    table_name = 'Prenotazioni'
    dynamodb = boto3.client('dynamodb')

    name = event['sessionState']['intent']['slots']['NomePrenotato']['value']['interpretedValue'].lower()
    surname = event['sessionState']['intent']['slots']['CognomePrenotato']['value']['interpretedValue'].lower()

    response = dynamodb.scan(
        TableName=table_name,
        FilterExpression='nome = :name and cognome = :surname',
        ExpressionAttributeValues={
            ':name': {'S': name},
            ':surname': {"S": surname},
        },
    )

    # Estrai gli elementi dalla risposta
    items = response.get('Items', [])
    response_text = ""

    if not items:
        return prepare_response_close(event, f"Nessuna prenotazione a nome {name} {surname}")

    response_text = f"Hai le seguenti prenotazioni:\n"
    for item in items:
        response_text += f"Codice {item['ID']['N']} - {item['giorno_prenotazione']['S']}: {item['ora_prenotazione']['S']}\n"

    return prepare_response_close(event, response_text)
```

Figura 3.65: Funzione `show_reservation` di `BotHandler`

remove_reservation

Questa funzione preleva dallo slot il valore del codice della prenotazione e, attraverso l'API `get_item` di `boto3`, ottiene nella variabile `response` l'oggetto che identifica quella prenotazione. Se tale oggetto esiste, con `delete_item` viene rimosso dalla tabella (Figura 3.66).

```python
def remove_reservation(event):
    dynamodb_resource = boto3.resource("dynamodb")
    table_name = 'Prenotazioni'
    reservation_table = dynamodb_resource.Table(table_name)

    id = event['sessionState']['intent']['slots']['ID']['value']['interpretedValue']

    response = reservation_table.get_item(
        Key={
            "ID": int(id),
        }
    )

    if "Item" not in response:
        return prepare_response(event,f"Nessuna prenotazione con codice {id} trovata")

    reservation_table.delete_item(
        Key={
            "ID": int(id),
        }
    )

    return prepare_response_close(event,f"Prenotazione {id} cancellata!")
```

Figura 3.66: Funzione `remove_reservation` di `BotHandler`

find_tdee

Questa funzione permette di calcolare il fabbisogno calorico giornaliero (TDEE) sulla base delle informazioni fornite dall'utente. Nella prima parte vengono raccolti i valori degli slot, e successivamente, attraverso delle valutazioni, viene applicata una formula per calcolare il BMR (Basal Metabolic Rate). In seguito, questo BMR viene moltiplicato per un fattore che indica il livello di attività fisica per ottenere il TDEE. Il valore ottenuto viene poi restituito sotto forma di messaggio all'utente (Figura 3.67).

```python
def find_tdee(event):
    tdee = None
    target = event['sessionState']['intent']['slots']['Target']['value']['interpretedValue']
    height = event['sessionState']['intent']['slots']['Height']['value']['interpretedValue']
    weight = event['sessionState']['intent']['slots']['Weight']['value']['interpretedValue']
    age = event['sessionState']['intent']['slots']['Age']['value']['interpretedValue']
    sex = event['sessionState']['intent']['slots']['SexType']['value']['interpretedValue']
    activity_level = event['sessionState']['intent']['slots']['ActivityLevelType']['value']['interpretedValue']
    activity_factor = None

    if(sex == "Maschio"):
        bmr = 88.362 + (13.397*int(weight)) + (4.799*int(height)) - (5.677*int(age))
    else:
        bmr = 447.593 + (9.247*int(weight)) + (3.098*int(height)) - (4.330*int(age))
    match activity_level:
        case "Sedentario":
            activity_factor = 1.2
        case "Leggermente attivo":
            activity_factor = 1.375
        case "Moderatamente attivo":
            activity_factor = 1.55
        case "Molto attivo":
            activity_factor = 1.725

    tdee = int(bmr * float(activity_factor))
    msgText = "Sulla base dei dati forniti e' risultato che il tuo TDEE e' di "+str(tdee)+"Kcal."
    match target:
        case "dimagrire":
            msgText+="\nQuindi ti consiglio di strutturare un piano alimentare in modo da assumere MENO KCAL rispetto al tuo TDEE."
        case "aumentare la massa magra":
            msgText+="\nQuindi ti consiglio di strutturare un piano alimentare in modo da assumere PIU' KCAL rispetto al tuo TDEE."
        case "mantenere la condizione attuale":
            msgText+="\nQuindi ti consiglio di strutturare un piano alimentare in modo da assumere LE STESSE KCAL rispetto al tuo TDEE."

    msgText2 = 'Se hai bisogno di altri consigli digita "Consigli".'
    return prepare_response(event,msgText,msgText2)
```

Figura 3.67: Funzione `find_tdee` di `BotHandler`

prepare_response e prepare_response_close

Queste due funzioni hanno il compito di adattare le risposte al formato che Amazon Lex si aspetta di ricevere (Figura 3.68). In particolare, deve essere specificata l'azione da svolgersi in seguito all'invio della risposta; tale azione può essere una tra le seguenti:

- *ConfirmIntent:* l'azione successiva consiste nel chiedere all'utente se l'intento è completo e pronto per essere soddisfatto. Questa è una domanda sì/no, ad esempio «Effettuare l'ordine?»
- *Close:* indica che non ci sarà una risposta da parte dell'utente. Ad esempio, la dichiarazione «Il tuo ordine è stato effettuato» non richiede una risposta.
- *Delegate:* l'azione successiva è determinata da Amazon Lex.
- *ElicitIntent:* l'azione successiva consiste nel determinare l'intento che l'utente desidera soddisfare.
- *ElicitSlot:* l'azione successiva consiste nell'ottenere un valore di slot dall'utente.

```python
def prepare_response_close(event, msgText):
    response = {
        "sessionState": {
            "dialogAction": {
                "type": "Close"
            },
            "intent": {
                "name": event['sessionState']['intent']['name'],
                "state": "Fulfilled"
            }
        },
        "messages": [
            {
                "contentType": "PlainText",
                "content": msgText
            }
        ]
    }

    return response
```

```python
def prepare_response(event, msgText,msgText2):
    response = {
        "sessionState": {
            "dialogAction": {
                "type": "Close"
            },
            "intent": {
                "name": event['sessionState']['intent']['name'],
                "state": "Fulfilled"
            }
        },
        "messages": [
            {
                "contentType": "PlainText",
                "content": msgText
            },
            {
                "contentType": "PlainText",
                "content": msgText2
            }
        ]
    }

    return response
```

Figura 3.68: Funzioni `prepare_response` di `BotHandler`

3.5.4 Deploy sulla piattaforma Telegram

Come fase finale vi è la distribuzione del chatbot su una piattaforma di messaggistica, in particolare Telegram. In questa ultima sottosezione descriveremo nel dettaglio tutto il procedimento per implementare tale funzionalità, illustrando le tecnologie usate. Prima di scendere nel dettaglio, nella Figura 3.69 riportiamo una panoramica dell'architettura AWS utilizzata per effettuare la comunicazione tra Telegram e Amazon Lex.

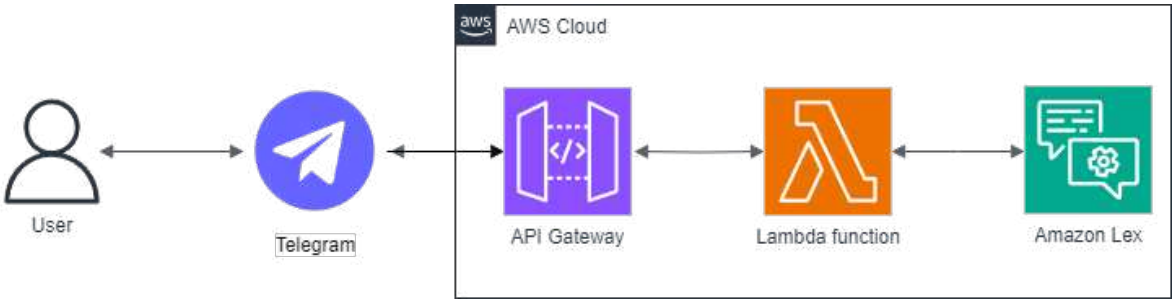


Figura 3.69: Architettura dei servizi AWS per il deploy su Telegram

Passiamo, ora, ad approfondire le varie fasi di sviluppo.

Creazione del bot Telegram

Come primo passaggio è stato necessario avviare una chat con il BotFather di Telegram per poter creare un proprio bot all'interno della piattaforma. Avvenuta la creazione, lo step successivo è stato quello di far generare al BotFather un'API Token, ovvero una stringa di codici che contiene dati completi che identificano un utente specifico. Ciò consente al server di autenticare le richieste dell'utente in chiamata e di convalidare l'estensione dell'utilizzo dell'API (Figura 3.70).

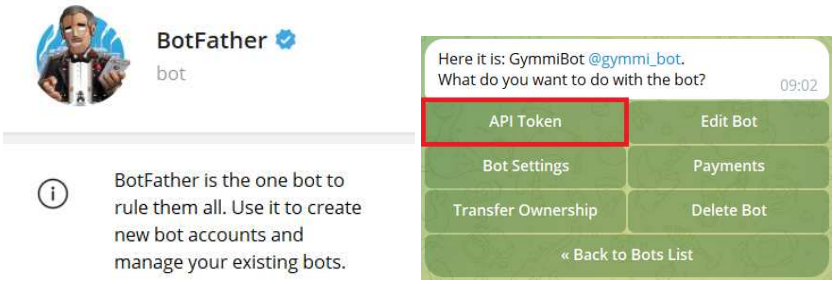


Figura 3.70: Interfaccia Telegram per la creazione del bot

Creazione e integrazione dell'API

Una volta creato il bot, è stata realizzata un'API HTTP su Amazon API Gateway per poter mettere in comunicazione Telegram e la funzione Lambda, che a sua volta comunica con il chatbot di Amazon Lex. In termini di configurazione, l'API ha una sola route che indirizza i messaggi in entrata verso l'integrazione specificata, ossia la funzione Lambda (Figura 3.71).

Creazione e configurazione della funzione Lambda

Come ultima fase occorre definire il "core" dell'architettura, ovvero una funzione Lambda che permetta di accogliere messaggi in ingresso da Telegram (Amazon Lex) e inviare messaggi

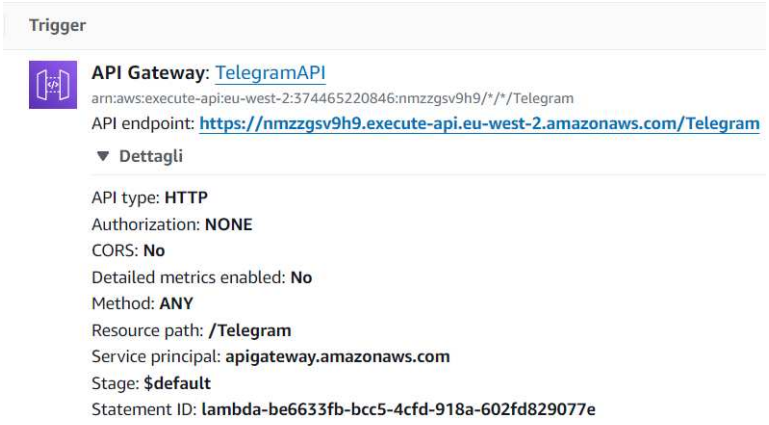


Figura 3.71: Configurazione dell'API

in uscita ad Amazon Lex (Telegram). L'invocazione di questa funzione è possibile poiché l'API è stata impostata come trigger della funzione; questo significa che non appena l'API identifica del traffico HTTP in entrata o in uscita alla funzione, esso viene prelevato dalla funzione stessa ed elaborato (Figura 3.72).

Figura 3.72: Diagramma della funzione Lambda

Approfondiamo ancora di più la funzione studiandone il codice per capirne il funzionamento (Figura 3.73).

Innanzitutto è stata utilizzata `Request`, una libreria HTTP di Python per l'invio di messaggi a Telegram tramite la sua API. Dopodiché abbiamo due passaggi importanti: il primo è quello della creazione della connessione con il servizio Amazon Lex tramite `boto3`; il secondo consiste nel salvataggio dell'API Token fornita dal BotFather di Telegram. Passando alla funzione `lambda_handler` notiamo l'utilizzo dell'argomento `event`, che contiene i messaggi inviati da Telegram, che vengono, estrapolati e salvati in una variabile locale. Tale variabile viene valutata, poiché all'avvio della chat con il bot esso invia automaticamente il messaggio `/start` che non è comprensibile da Amazon Lex; pertanto, esso, viene trasformato in una frase di esempio che attiva l'intent di benvenuto. Successivamente abbiamo le due funzioni che operano lo scambio di messaggi tra le due piattaforme (Figura 3.74). Analizziamole nel dettaglio.

La funzione `send_to_lex` riceve come argomento il messaggio ricevuto da Telegram; tramite l'API `recognize_text` di `boto3`, tale messaggio viene inviato al chatbot che restituirà la risposta ad esso. Invece, la funzione `send_telegram_message` riceve come argomenti `chat_id` e `text`, che sono, rispettivamente, il codice della chat Telegram con il bot e il messaggio da inviare. Quest'ultimo viene incluso nel paylaod della richiesta HTTP dell'API di Telegram e, alla fine, la libreria di Python si occuperà del suo invio. Di seguito è possibile vedere un esempio del funzionamento del chatbot di Amazon Lex su Telegram (Figura 3.75).

```python
import json
import os
import boto3
from urllib.parse import urlencode
from urllib.request import Request, urlopen

lex_client = boto3.client('lexv2-runtime',region_name='eu-west-2')
TELEGRAM_BOT_TOKEN = os.environ['TOKEN']

def send_to_lex(user_message):...
def send_telegram_message(chat_id, text):...

def lambda_handler(event, context):

    body = json.loads(event['body'])
    chat_id = body['message']['chat']['id']
    user_message = body['message']['text']

    # Se sto avviando la chat triggero l'intent benvenuto
    if user_message == "/start":
        user_message = "Ciao"

    # Invio il messaggio da telegram a lex
    lex_response = send_to_lex(user_message)
    print(lex_response['messages'])
    lex_message = lex_response['messages']

    # Invio il messaggio di lex a telegram
    send_telegram_message(chat_id, lex_message)

    return {
        'statusCode': 200,
        'body': json.dumps({'message': 'Message processed successfully'})
    }
```

Figura 3.73: Codice della funzione Lambda per l'integrazione con Telegram

```python
def send_to_lex(user_message):
    lex_response = lex_client.recognize_text(
        botId='OP9HV8MN9P',
        botAliasId='TSTALIASID',
        localeId='it_IT',
        sessionId='100',
        text=user_message
    )
    return lex_response
```

```python
def send_telegram_message(chat_id, text):
    response = ""

    for element in text:
        response += element['content']+"\n"

    telegram_url = f'https://api.telegram.org/bot{TELEGRAM_BOT_TOKEN}/sendMessage'
    payload = {
        'chat_id': chat_id,
        'text': response
    }

    request = Request(telegram_url, urlencode(payload).encode())
    out = urlopen(request).read().decode()
```

Figura 3.74: Codice delle funzioni che gestiscono il traffico di messaggi

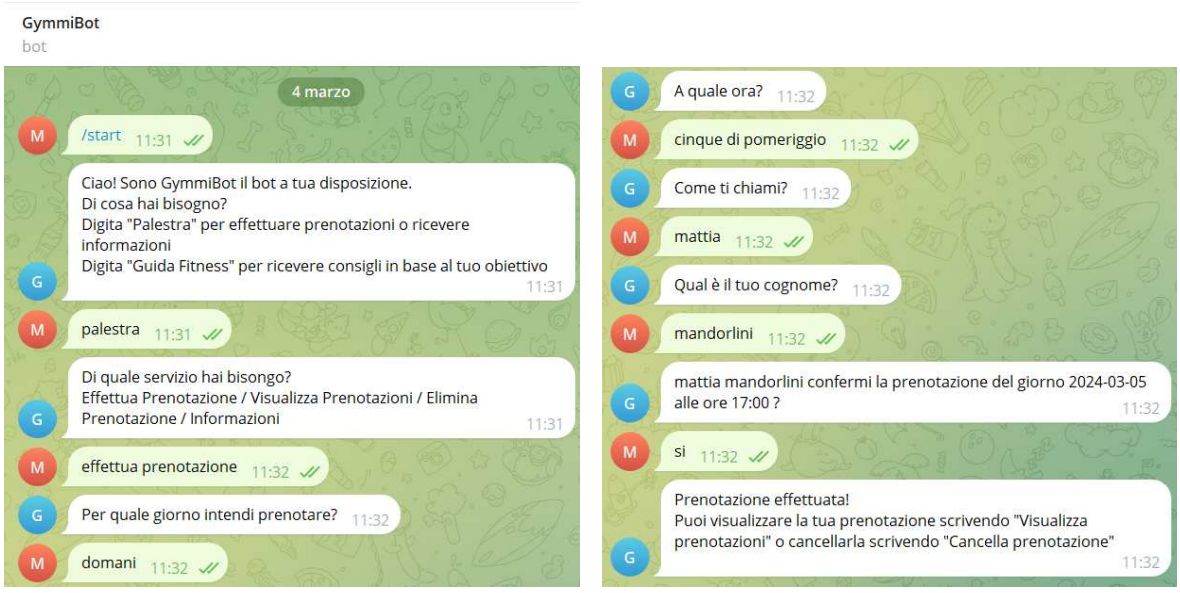


Figura 3.75: Test di esempio del chatbot su Telegram

CAPITOLO 4

Esperienza sul riconoscimento di immagini

In questo capitolo tratteremo un'altra tra le più diffuse e importanti applicazioni dell'Intelligenza Artificiale: il riconoscimento di immagini. Per fare ciò, cominceremo la discussione introducendo il concetto di "Computer vision" e parleremo del suo funzionamento. Dopodiché, passeremo ad approfondire il riconoscimento di immagini, che è un caso particolare di utilizzo della computer vision. Alla fine, analizzeremo dettagliatamente il secondo case study affrontato, parlando dell'obiettivo, dell'architettura e descrivendo le varie tecnologie usate.

4.1 Computer vision

La visione artificiale rappresenta un campo dell'Intelligenza Artificiale che sfrutta il machine learning e le reti neurali per istruire computer e sistemi a estrarre informazioni significative da immagini digitali, video e altri input visivi. Questi sistemi sono in grado di fornire raccomandazioni o intraprendere azioni nel momento in cui individuano difetti o problematiche. Se l'IA permette ai computer di ragionare, la visione artificiale consente loro di vedere, osservare e comprendere il mondo circostante.

La visione artificiale funziona sostanzialmente come la vista umana, con la differenza che gli esseri umani hanno un vantaggio iniziale. La vista umana, infatti, beneficia di una vita intera di contesto per imparare a distinguere gli oggetti, valutarne la distanza, individuare movimenti o eventuali anomalie nelle immagini. L'addestramento della visione artificiale comporta l'insegnamento alle macchine di eseguire tali funzioni, ma in un tempo molto più breve, utilizzando telecamere, dati e algoritmi, anziché retine, nervi ottici e una corteccia visiva. Grazie a sistemi addestrati per ispezionare prodotti o monitorare asset di produzione, è possibile analizzare migliaia di prodotti o processi al minuto, individuando difetti o problemi impercettibili e superando rapidamente le capacità umane. La visione artificiale trova applicazione in una vasta gamma di settori, dall'energia e i servizi pubblici alla produzione e all'industria automobilistica, e il mercato continua a crescere costantemente.

Nonostante la tecnologia per l'elaborazione delle informazioni visive esista da tempo, in passato gran parte del processo richiedeva l'intervento umano, consumava molto tempo ed era soggetto a errori. Per esempio, la creazione di un sistema di riconoscimento facciale richiedeva agli sviluppatori di etichettare manualmente migliaia di immagini con punti dati chiave, come la larghezza del ponte nasale e la distanza tra gli occhi. Automatizzare queste attività richiedeva una considerevole potenza di calcolo, poiché i dati delle immagini sono non strutturati e complessi da organizzare per i computer. Di conseguenza, le applicazioni di visione erano costose e al di fuori della portata della maggior parte delle organizzazioni.

Oggi, grazie ai progressi nel campo combinati con un notevole aumento della potenza di calcolo, sia la scala che la precisione dell'elaborazione dei dati delle immagini sono notevolmente migliorate. I sistemi di visione artificiale basati sulle risorse di cloud computing sono ora accessibili a tutti.

4.1.1 Funzionamento della computer vision

I sistemi di visione artificiale utilizzano la tecnologia dell'Intelligenza Artificiale per imitare le capacità del cervello umano responsabili del riconoscimento e della classificazione degli oggetti. La visione artificiale richiede un'enorme quantità di dati che analizza ripetutamente fino a distinguere le differenze e, alla fine, riconoscere le immagini. Ad esempio, per addestrare un computer a riconoscere pneumatici automobilistici, è necessario alimentarlo con vaste quantità di immagini di pneumatici e oggetti correlati per imparare le differenze e riconoscere uno, specialmente senza difetti.

Due tecnologie essenziali vengono utilizzate per raggiungere questo obiettivo:

- *Deep learning:* attraverso modelli algoritmici, i computer possono apprendere il contesto dei dati visivi. Quando il modello riceve dati sufficienti, il computer li "osserva" e impara autonomamente a distinguere un'immagine dall'altra. Gli algoritmi consentono alla macchina di apprendere autonomamente anziché essere programmata per riconoscere un'immagine.
- *Reti neurali convoluzionali (CNN):* attraverso modelli di deep learning, i computer possono "guardare" suddividendo le immagini in pixel a cui vengono assegnati tag o etichette. Utilizzando queste etichette, eseguono convoluzioni, un'operazione matematica che combina due funzioni per produrne una terza. La rete neurale esegue convoluzioni e controlla l'accuratezza delle sue previsioni in una serie di iterazioni finché le previsioni iniziano a diventare vere. A quel punto, riconosce o "vede" le immagini in modo simile agli esseri umani. Come un umano che distingue un'immagine da lontano, una CNN prima individua i bordi netti e le forme semplici, poi completa le informazioni durante le iterazioni delle sue previsioni. Una CNN è impiegata per comprendere singole immagini, mentre una rete neurale ricorrente (RNN) viene utilizzata in modo analogo per applicazioni video, aiutando i computer a comprendere come le immagini in una serie di fotogrammi siano correlate tra loro.

4.1.2 Applicazioni della computer vision

Lo sviluppo delle tecnologie di deep learning ha consentito la creazione di modelli di computer vision più accurati e complessi. Con la diffusione di queste tecnologie, l'incorporazione di applicazioni di visione artificiale sta diventando sempre più vantaggiosa. Di seguito sono riportati alcuni casi d'uso; inoltre, in Figura 4.1 è possibile osservare come e dove la computer vision viene utilizzata.

Sicurezza e protezione

I governi e le imprese impiegano la visione artificiale per potenziare la sicurezza delle risorse, dei siti e delle strutture. Ad esempio, telecamere e sensori monitorano gli spazi pubblici, i siti industriali e gli ambienti ad alta sicurezza, fornendo avvisi automatici in caso di eventi insoliti, come l'ingresso di individui non autorizzati in aree riservate.

In modo analogo, la visione artificiale può migliorare la sicurezza personale sia a casa che sul luogo di lavoro. Per esempio, può monitorare una vasta gamma di problematiche

Figura 4.1: Casi d'uso della computer vision

legate alla sicurezza, come lo streaming in tempo reale per la sorveglianza degli animali domestici a casa o l'identificazione dei visitatori e dei pacchi consegnati tramite telecamere in diretta. Sul posto di lavoro, questo monitoraggio comprende l'uso appropriato di dispositivi di protezione individuale da parte dei lavoratori, il segnalamento ai sistemi di allarme e la generazione di rapporti.

Efficienza operativa

La visione artificiale può analizzare immagini ed estrarre metadati per la Business Intelligence[1], creando nuove opportunità di profitto ed efficienze operative. Ad esempio, può:

- identificare automaticamente i difetti di qualità prima che i prodotti lascino la fabbrica;
- rilevare i problemi di manutenzione e sicurezza delle macchine;
- analizzare le immagini dei social media per scoprire tendenze e modelli di comportamento dei clienti;
- autenticare i dipendenti grazie al riconoscimento facciale automatico.

Sanità

L'assistenza sanitaria è uno dei settori leader per l'applicazione della tecnologia di visione artificiale. In particolare, l'analisi delle immagini mediche crea una visualizzazione di organi e tessuti per aiutare i medici a effettuare diagnosi rapide e accurate, con conseguenti migliori risultati dei trattamenti e una migliore aspettativa di vita. Esempi di applicazione della computer vision nella sanità sono i seguenti:

- rilevazione di tumori mediante analisi di nei e lesioni cutanee;
- analisi radiografica automatica;
- scoperta dei sintomi dalle risonanze magnetiche.

[1] La "Business Intelligence" (BI) è un termine utilizzato per descrivere il processo di raccolta, analisi e presentazione di informazioni aziendali significative e utili per supportare le decisioni strategiche e operative all'interno di un'organizzazione.

Veicoli a guida autonoma

La tecnologia dei veicoli autonomi utilizza la visione artificiale per riconoscere immagini in tempo reale e creare mappe 3D da più telecamere montate sul veicolo autonomo. Può analizzare le immagini e identificare altri utenti della strada, segnali stradali, pedoni o ostacoli. Nei veicoli semiautonomi, la visione artificiale utilizza il machine learning per monitorare il comportamento del conducente. Ad esempio, cerca segni di distrazione, affaticamento e sonnolenza in base alla posizione della testa del conducente, al tracciamento degli occhi e al movimento della parte superiore del corpo. Se la tecnologia rileva determinati segnali di pericolo, avvisa il conducente e riduce la possibilità di un incidente di guida.

4.2 Il riconoscimento di immagini

Il riconoscimento di immagini è una tecnologia che consente di identificare vari elementi all'interno di immagini digitali, tra cui luoghi, loghi, persone e oggetti. Mentre gli esseri umani possono facilmente riconoscere diverse immagini, per un computer questo compito può essere più complesso. Le immagini digitali sono composte da pixel, ciascuno con una rappresentazione numerica per l'intensità o il livello di grigio (Figura 4.2). Il computer interpreta le immagini come valori numerici dei pixel e, per riconoscerle, deve individuare i modelli e le regolarità in questi dati. Questa tecnologia ha numerose applicazioni pratiche in settori come produzione, sanità, vendita al dettaglio, agricoltura e sicurezza, migliorando la qualità del controllo, facilitando la diagnosi medica, ottimizzando i processi commerciali e contribuendo alla sicurezza e sorveglianza.

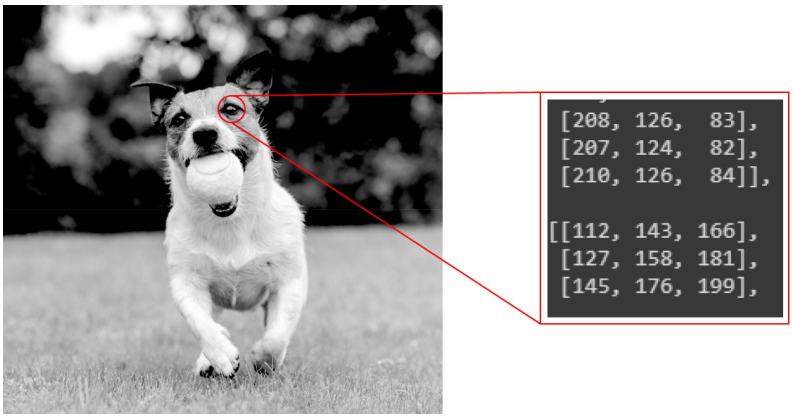


Figura 4.2: Digitalizzazione delle immagini

Il riconoscimento di immagini, nel contesto della visione artificiale, costituisce una sfida per i computer, nonostante sia un compito semplice per il cervello umano. Svariate metodologie vengono impiegate per l'elaborazione delle immagini, tra cui i modelli di deep learning e machine learning, la cui selezione dipende dal caso d'uso. Di solito, il riconoscimento di immagini coinvolge la costruzione di reti neurali profonde che analizzano i singoli pixel dell'immagine. Il processo di riconoscimento di immagini inizia con la raccolta di un vasto dataset di immagini etichettate. Queste immagini, opportunamente contrassegnate, fungono da materiale di addestramento per gli algoritmi, consentendo loro di apprendere e distinguere i modelli e le caratteristiche presenti in una varietà di contesti visivi. Successivamente, le immagini devono essere sottoposte a una fase di pre-elaborazione, durante la quale vengono eliminate eventuali fonti di disturbo, come rumore, distorsioni o altri artefatti, che potrebbero influenzare negativamente il processo di riconoscimento. Ciò può implicare operazioni come

il ridimensionamento, il ritaglio o l'aggiustamento del contrasto e della luminosità. Una volta preparate, le immagini subiscono l'estrazione delle caratteristiche, un passo fondamentale che implica l'identificazione e l'isolamento delle parti salienti dell'immagine. Queste caratteristiche sono essenziali per consentire agli algoritmi di distinguere tra oggetti o categorie differenti. Segue, quindi, l'addestramento del modello, nel quale l'algoritmo, ormai istruito sulle caratteristiche estratte, apprende a riconoscere e categorizzare gli oggetti presenti nelle immagini. Dopo l'addestramento, il modello viene testato e valutato su un dataset separato per valutare la sua accuratezza e le sue prestazioni. Questa fase di test consente di individuare eventuali errori o debolezze nel modello che devono essere affrontati e corretti. Una volta che il modello è stato testato e convalidato, può essere implementato per classificare nuove immagini con precisione, rappresentando l'ultimo passaggio nel processo di riconoscimento di immagini. Nella Figura 4.3 vengono mostrati i passaggi fondamentali del processo di riconoscimento delle immagini.

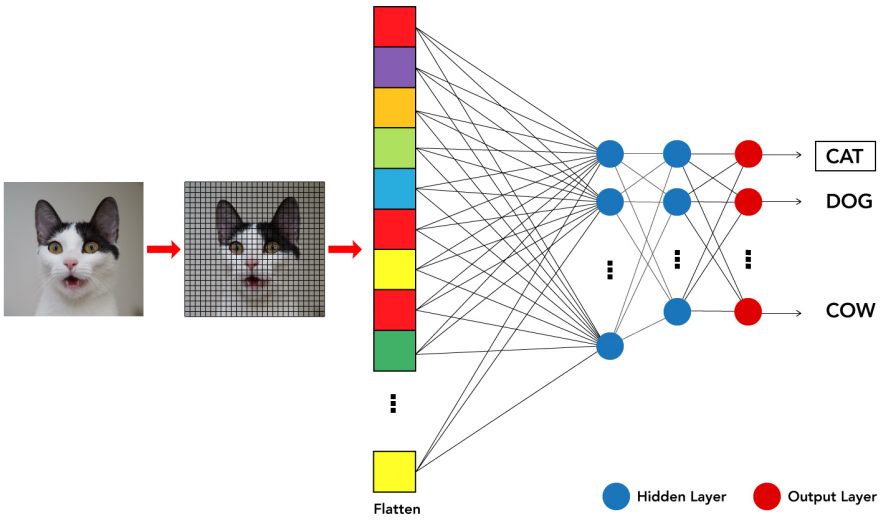


Figura 4.3: Funzionamento del riconoscimento di immagini

4.2.1 Il futuro del riconoscimento di immagini

Il futuro del riconoscimento di immagini si presenta ricco di promesse e offre un vasto campo di applicazioni in molteplici settori industriali. Uno degli orientamenti principali nello sviluppo è l'integrazione della tecnologia di riconoscimento di immagini con l'Intelligenza Artificiale e il machine learning, consentendo alle macchine di acquisire conoscenza dall'esperienza e di perfezionare la loro precisione nel tempo. L'adozione di soluzioni basate sul cloud nel campo del riconoscimento di immagini conferisce alle aziende la flessibilità di implementare rapidamente tali tecnologie senza la necessità di infrastrutture complesse. Inoltre, il riconoscimento di immagini gioca un ruolo fondamentale nello sviluppo dei veicoli autonomi, permettendo loro di analizzare l'ambiente circostante e di prevenire incidenti grazie all'identificazione tempestiva di ostacoli e pedoni. Queste tecnologie trovano applicazione anche in settori quali gli occhiali intelligenti, la realtà aumentata e la previsione del comportamento dei consumatori, apportando significative opportunità di miglioramento e innovazione sia per le imprese che per la società nel suo complesso.

4.3 Descrizione del case study

In questa sezione esamineremo in dettaglio il secondo case study sviluppato che riguarda l'implementazione del riconoscimento facciale per creare un sistema di accesso basato sulla scansione dei volti, dove gli utenti possono ottenere l'autorizzazione per accedere a un'area solo se vengono riconosciuti come autorizzati.

4.3.1 Architettura dell'applicazione

Inizieremo fornendo un'introduzione generale sul funzionamento dei diversi servizi e tecnologie impiegati per realizzare l'obiettivo iniziale, riservando i dettagli implementativi alle sottosezioni successive. Nella Figura 4.4 è possibile osservare l'architettura dei servizi AWS per la realizzazione dell'applicazione.

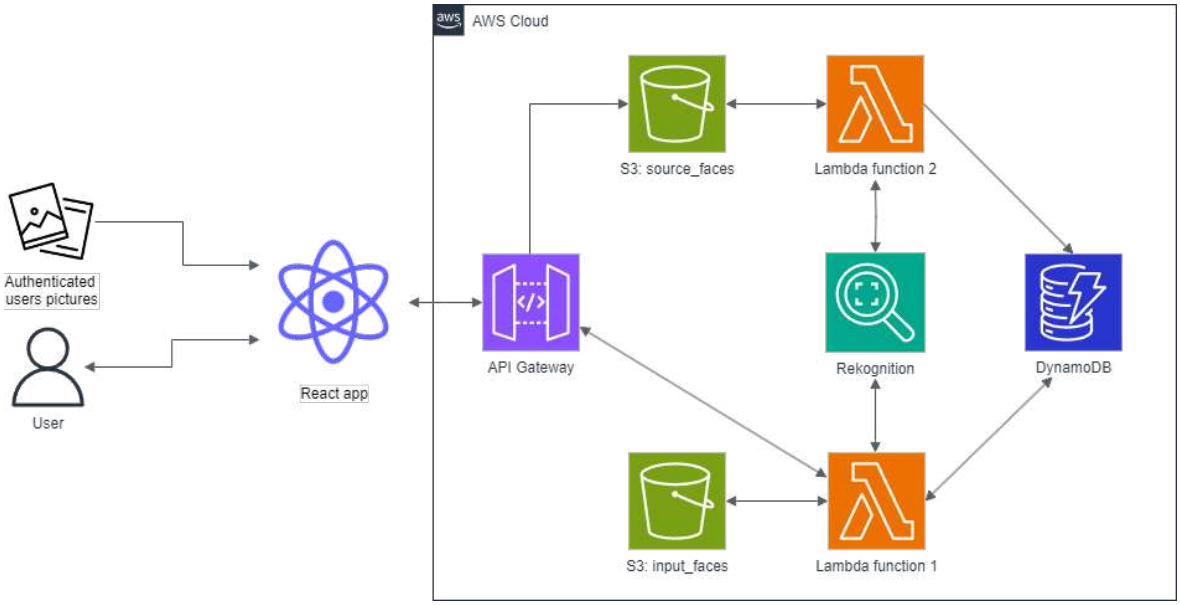


Figura 4.4: Architettura dell'applicazione di riconoscimento facciale

L'integrazione di un'interfaccia grafica è stata realizzata per semplificare l'interazione dell'utente con il servizio di riconoscimento facciale. Tale interfaccia mette a disposizione una sezione che consente di testare l'applicazione con maggiore facilità, permettendo di caricare le immagini dei volti degli utenti autorizzati e di accedere alla webcam del dispositivo per scattare un'istantanea del proprio volto. Per consentire all'applicazione React di comunicare con i servizi AWS, è stata creata una REST API con due endpoint, uno per ciascuna funzionalità. Il primo endpoint consente di caricare direttamente un'immagine su un bucket S3, il che attiva una funzione Lambda incaricata di recuperare l'immagine, di indicizzarla utilizzando un'API di Amazon Rekognition e di salvare l'indice corrispondente in una tabella di DynamoDB. Il secondo endpoint, invece, viene utilizzato quando l'utente scatta un'istantanea del proprio volto e permette di attivare una funzione Lambda incaricata di verificare l'autenticazione facciale confrontando il volto acquisito con quelli caricati nel bucket di S3.

4.3.2 Tecnologie di front-end

Iniziamo l'analisi dell'applicazione partendo dal lato client, e quindi, con React JS, una libreria open-source, front-end, JavaScript per la creazione di interfacce utente. Nella Figura 4.5 viene mostrata l'interfaccia grafica realizzata.

Figura 4.5: Interfaccia utente dell'applicazione di riconoscimento facciale

Nella parte sinistra troviamo la sezione dedicata all'attivazione della videocamera e alla cattura delle istantanee, mentre nella parte destra è presente la sezione per caricare nuove immagini nel bucket S3.

Iniziamo mostrando, nella Figura 4.6, il codice relativo al caricamento di un'immagine sul bucket. La funzione `loadImage` viene chiamata al click del pulsante "Carica immagine" e riceve come parametro l'oggetto che rappresenta l'evento del click del mouse sul pulsante. Attraverso l'istruzione `fetch`, viene effettuata una richiesta HTTP all'endpoint dell'API, passando l'immagine da caricare come parametro. Nell'header della richiesta viene specificato il formato dei dati inviato tramite il metodo `PUT`, mentre nel `body` è contenuta l'immagine stessa.

Mediante il metodo `then`, è possibile definire una procedura da eseguire nel caso in cui la richiesta abbia esito positivo. In questo caso, viene mostrato a video un messaggio a seconda dello stato della risposta HTTP.

```
const loadImage = (e) => {
  e.preventDefault();
  fetch(`https://7zkgxx1i53.execute-api.eu-west-2.amazonaws.com/dev/source-faces/${image.name}`, {
    method: 'PUT',
    headers: {
      'Content-Type': 'image/jpeg'
    },
    body: image
  }).then((res) => {
    if (res.status === 200) {
      setUploadStatus('Caricamento avvenuto con successo!');
      setUpload(true);
    } else {
      setUploadStatus('Caricamento fallito!');
      setUpload(false);
    }
    inputfileRef.current.value = '';
  })
}
```

Figura 4.6: Codice dell'applicazione React per il caricamento delle immagini nel bucket S3

Passiamo, ora, ad analizzare il codice della sezione di sinistra (Figura 4.7). Innanzitutto, è stata utilizzata la libreria `react-webcam` per una gestione ottimale della videocamera e delle immagini. Ora, consideriamo la funzione principale `captureAndSend` e, in particolare, consideriamo l'uso del React Hook `useCallback`, che fa il caching della funzione e la restituisce solo se le dipendenze specificate cambiano. In questo modo, la funzione viene ricalcolata solo quando è strettamente necessario, ottimizzando le prestazioni dell'applicazione. Quando l'utente clicca sul pulsante "Cattura immagine", avviene la chiamata alla funzione, che utilizza il metodo `getScreenshot` della libreria per catturare uno screenshot della schermata video. Successivamente, l'immagine viene elaborata e trasformata in un oggetto BLOB (Binary Large Object) per consentire l'elaborazione da parte di Amazon Rekognition. Questo oggetto viene poi utilizzato come contenuto di una richiesta HTTP all'endpoint dell'API per il caricamento dell'immagine su un bucket specifico, dedicato all'archiviazione delle istantanee scattate. Una volta completata la richiesta, la risposta viene gestita attraverso il metodo `then`, che chiama la funzione `auth` per determinare se l'autenticazione è avvenuta con successo o meno.

```
const captureAndSend = useCallback(() => {
        const imageSrc = webcamRef.current.getScreenshot();
        setImgSrc(imageSrc);
        // rimuovo il prefisso del formato base64 , '');
        // Converto base64 a Uint8Array
        const uint8Array = new Uint8Array(atob(base64WithoutPrefix).split('').map(char => char.charCodeAt(0)));

        const blob = new Blob([uint8Array], { type: 'image/jpeg' });

        const userImageName = uuid.v4();//genero un id da associare all'immagine

        //carico l'immagine nel bucket input-faces chiamando l'endpoint dell'API Gateway
        fetch(`https://7zkgxx1i53.execute-api.eu-west-2.amazonaws.com/dev/input-faces/${userImageName}.jpg`, {
            method: 'PUT',
            headers: {
                'Content-Type': 'image/jpeg'
            },
            body: blob
        }).then(async () => {
            //se tutto va bene controllo se l'utente è autorizzato
            const response = await auth(userImageName);
            if (response.Message === 'Success') {
                setIsAuth(true);
                setUploadResultMessage('Autenticazione avvenuta con successo!');
            } else {
                setIsAuth(false);
                setUploadResultMessage('Autenticazione fallita!');
            }
        }).catch(error => {
            setIsAuth(false);
            setUploadResultMessage("Errore: impossibile procedere con l'autenticazione");
            console.error(error);
        })
        setCaptureDisabled(true);
        setRetryDisabled('');
    }, [webcamRef]);
```

Figura 4.7: Codice dell'applicazione React per il caricamento delle istantanee

La funzione `auth` (Figura 4.8) riceve come parametro un identificativo univoco universale (uuid) che rappresenta il nome con cui viene salvata l'immagine nel bucket. Il compito principale di questa funzione è prendere questo parametro e inviarlo come parametro URL nella richiesta HTTP al secondo endpoint dell'API, che si occuperà dell'interazione con Amazon Rekognition. Quest'ultimo effettuerà il matching con le altre immagini degli utenti autorizzati e ritornerà il risultato dell'operazione.

```
const auth = async (userImageName) => {
        const requestURL = 'https://7zkgxx1i53.execute-api.eu-west-2.amazonaws.com/dev/inputface?'+
        new URLSearchParams({ objectKey: `${userImageName}.jpg` });
        return await fetch(requestURL, {
            method: 'GET',
            headers: {
                'Accept': 'application/json',
                'Content-Type': 'application/json'
            }
        }).then(response => response.json())
            .then((data) => {
                return data;
            })
            .catch(error => console.error(error))
    }
```

Figura 4.8: Codice dell'applicazione React per la verifica dell'autenticazione

Infine, è stata realizzata una piccola funzione, mostrata in Figura 4.9, che permette di ripristinare lo stato iniziale dell'applicazione per consentire all'utente di effettuare una nuova istantanea.

```
const retake = () => {
        setImgSrc(null);
        setIsAuth(false);
        setUploadResultMessage('Scatta una tua foto per autenticarti');
        setCaptureDisabled('');
        setRetryDisabled(true);
    }
```

Figura 4.9: Codice dell'applicazione React per effettuare nuove istantanee

4.3.3 Tecnologie di back-end e API

Le elaborazioni dei dati e le procedure di gestione delle risposte eseguite lato client facilitano la struttura del back-end dell'applicazione. Attraverso AWS Lambda sono state implementate due funzioni che gestiscono le richieste HTTP provenienti dal front-end. Procediamo, ora, analizzando la funzione responsabile di indicizzare e memorizzare nell'apposito database le immagini caricate nel bucket `source-faces`, che riguarda gli utenti autorizzati. Tale funzione effettua le connessioni ai servizi AWS necessari, inclusi S3, Rekognition e DynamoDB, quindi rimane in attesa del caricamento di un'immagine nel bucket. Una volta attivato il trigger impostato (Figura 4.10), viene invocata la funzione `lambda_handler`. A questo punto, vengono estratti il nome del bucket e l'immagine caricata, in modo che l'API `index_faces` di Amazon Rekognition possa indicizzarla e, successivamente, possa salvare

tale indice in una tabella di DynamoDB tramite l'API `put_item`. Nel codice della funzione Lambda, riportato nella Figura 4.11, sono dettagliati i passaggi sopra descritti.

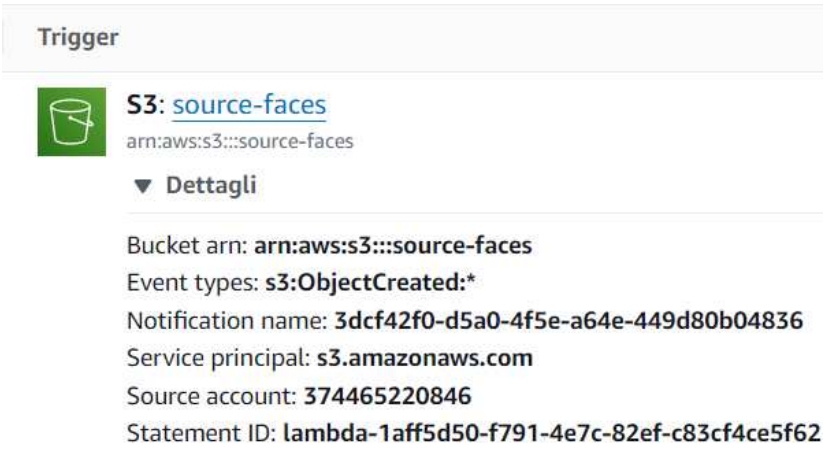


Figura 4.10: Impostazione del trigger S3 alla funzione Lambda `source-faces`

```python
import boto3

s3 = boto3.client('s3')
rekognition = boto3.client('rekognition', region_name='eu-west-2')
dynamodb = boto3.resource('dynamodb', region_name='eu-west-2')

dynamodbTableName = 'sourceFaces'
sourceFacesTable = dynamodb.Table(dynamodbTableName)

def lambda_handler(event, handler):
    bucket_name = event['Records'][0]['s3']['bucket']['name']
    key = event['Records'][0]['s3']['object']['key']

    try:
        response = index_source_faces(bucket_name, key)
        if response['ResponseMetadata']['HTTPStatusCode'] == 200:
            faceId = response['FaceRecords'][0]['Face']['FaceId']
            register_source_face(faceId)
    except Exception as e:
        print(e)
        raise e
```

```python
def index_source_faces(bucket, key):
    response = rekognition.index_faces(
        Image={
            'S3Object': {
                'Bucket': bucket,
                'Name': key
            }
        },
        CollectionId="faces"
    )
    return response

def register_source_face(facedId):
    sourceFacesTable.put_item(
        Item={
            'rekognitionid': facedId
        }
    )
```

Figura 4.11: Codice della funzione Lambda `source-faces`

Quanto alla seconda funzione, essa si dedica alla verifica dell'autenticazione. Come la funzione precedente, essa si connette inizialmente ai servizi AWS, quindi resta in attesa che l'utente scatti una foto per attivare il trigger impostato (Figura 4.12) e invocare la suddetta funzione. Quest'ultima estrae dall'URL l'identificatore univoco (UUID) dell'immagine, che viene utilizzato per recuperare l'immagine stessa dal bucket. Quest'ultima viene, poi, convertita in formato binario in modo che l'API `search_faces_by_image` di Amazon Rekognition possa analizzarla. In risposta all'API, viene restituito un array contenente informazioni su tutte le corrispondenze individuate dall'analisi facciale e, per ciascuna di queste corrispondenze, viene verificata la presenza di un'associazione con gli ID memorizzati nel database: in caso di corrispondenza, l'autenticazione è considerata riuscita; in caso contrario, l'utente non è autorizzato. In Figura 4.13 viene mostrato in dettaglio il codice della funzione `faces_auth`.

Infine, nella Figura 4.14, viene presentata la struttura dell'API. La risorsa `/{bucket}/{filename}` consente il caricamento di un file specifico nel bucket indicato, mentre la

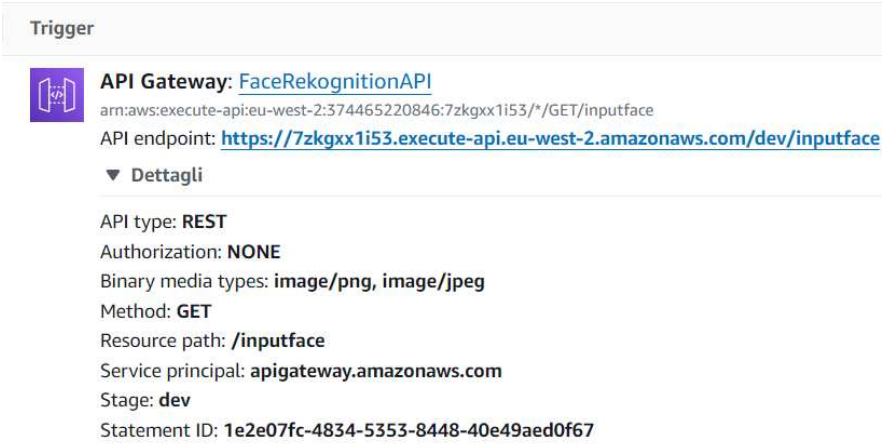


Figura 4.12: Impostazione del trigger API Gateway alla funzione Lambda `faces_auth`

```python
import boto3
import json

s3 = boto3.client('s3')
rekognition = boto3.client('rekognition', region_name='eu-west-2')
dynamodb = boto3.resource('dynamodb', region_name='eu-west-2')

dynamodbTableName = 'sourceFaces'
sourceFacesTable = dynamodb.Table(dynamodbTableName)
bucket_name = 'input-faces'

def lambda_handler(event, context):

    objectKey = event['queryStringParameters']['objectKey']
    image_bytes = s3.get_object(Bucket=bucket_name, Key=objectKey)['Body'].read()

    response = rekognition.search_faces_by_image(
        CollectionId='faces',
        Image={'Bytes': image_bytes}
    )

    for match in response['FaceMatches']:
        print(match['Face']['FaceId'], match['Face']['Confidence'])
        face = sourceFacesTable.get_item(
            Key={
                'rekognitionid': match['Face']['FaceId']
            }
        )
        if 'Item' in face:
            return build_response(200, {'Message': 'Success'})

    return build_response(401, {'Message': 'Utente non autorizzato'})
```

Figura 4.13: Codice della funzione Lambda `faces_auth`

risorsa `/inputface` attiva la funzione Lambda `faces_auth`. Entrambe le risorse sono dotate di default del metodo `OPTIONS`, che include gli header Access-Control-Allow-Origin, Access-Control-Allow-Methods e Access-Control-Allow-Headers. Questi header forniscono al browser indicazioni sulla concessione della richiesta effettiva e sui metodi e header consentiti, promuovendo una comunicazione sicura tra client e server all'interno del contesto di Cross-Origin Resource Sharing (CORS). Oltre al metodo `OPTIONS`, è possibile definirne degli altri; ad esempio, per la prima risorsa è stato configurato il metodo `PUT`, che consente di aggiornare o sostituire dati o risorse sul server. Per la seconda risorsa, invece, è stato definito

il metodo `GET`, utilizzato per richiedere e ottenere dati dal server.

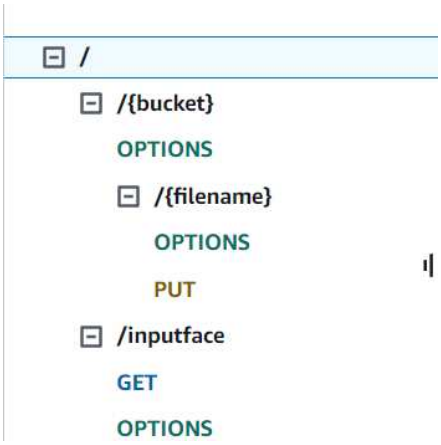


Figura 4.14: Struttura dell'API dell'applicazione

4.3.4 Storage dei dati

Come ultima fase della descrizione vogliamo motivare le scelte effettuate per quanto riguarda l'utilizzo dei servizi di storage. Amazon S3 consente di archiviare e recuperare qualsiasi quantità di dati, in qualunque momento e da ogni luogo tramite il Web. Esso è progettato per fornire una durabilità del 99,999999999% e una disponibilità pari al 99,99% per le proprie esigenze di storage nel cloud. Per questi motivi, la scelta di archiviare numerose immagini in un bucket S3 è estremamente vantaggiosa e sicura. Nella Figura 4.15 è possibile osservare gli oggetti immagazzinati nei due bucket S3.

Nome	Tipo	Ultima modifica	Dimensioni
110e8f03-8c87-4138-8bb7-c92f6e2cc2eb.jpg	jpg	22 Dec 2023 10:06:52 AM CET	190.2 KB
5c90b907-84d9-4181-83fd-219834c59ceb.jpg	jpg	22 Dec 2023 10:07:30 AM CET	191.0 KB
63c2c67b-2e98-468c-8e46-4019318fed90.jpg	jpg	06 Mar 2024 01:55:46 PM CET	165.9 KB
8d972463-dcf8-49a4-9977-3433ad908437.jpg	jpg	06 Mar 2024 01:55:57 PM CET	165.2 KB

Nome	Tipo	Ultima modifica	Dimensioni
mat.jpg	jpg	22 Dec 2023 10:07:26 AM CET	82.5 KB
spi.jpg	jpg	14 Mar 2024 03:17:33 PM CET	76.2 KB
tati.jpg	jpg	14 Mar 2024 03:17:33 PM CET	75.8 KB

Figura 4.15: Bucket S3 `source_faces` e `input_faces`

In aggiunta a S3 come strumento di archiviazione, viene utilizzata anche una tabella DynamoDB per memorizzare gli indici delle immagini generati da Rekognition. La scelta di questo tipo di archiviazione non è motivata da particolari esigenze, se non dalla sua semplicità e velocità di gestione dei dati. Nella Figura 4.16 è presente un'esposizione del contenuto della tabella.

rekognitionid *(Stringa)*
014b50e4-4b50-48c4-859a-806e88d6e9cf
29ee0717-7893-4a2b-a7f5-4a1ad4694a3c
17a0fc8e-8c94-4dc1-9ebb-46975bc2230c

Figura 4.16: Tabella DynamoDB `sourceFaces`

Come eventuale miglioramento futuro potrebbe essere utile memorizzare, oltre all'indice, il nome e il cognome degli utenti autorizzati.

4.3.5 Test dell'applicazione

I test dell'applicazione seguiranno questa sequenza di passaggi:

- Nel bucket che contiene le immagini degli utenti autorizzati, la mia foto sarà rimossa e verrà eseguito il primo test dell'applicazione. Il risultato di questo test sarà "Autenticazione fallita", poiché l'API `search_faces_by_image` non troverà corrispondenze (Figura 4.17).

Figura 4.17: Risultato del primo test

- Una mia immagine sarà caricata dalla sezione apposita e inserita nel bucket menzionato in precedenza (Figura 4.18).

Figura 4.18: Risultato del secondo test

- Si procederà con un ultimo test, il cui esito sarà "Autenticazione avvenuta con successo", poiché ora, l'API, riconoscerà il mio volto tra quelli memorizzati nel sistema (Figura 4.19).

Figura 4.19: Risultato del terzo test

CAPITOLO 5

Esperienza sulla sentiment analysis

In questo capitolo discuteremo la terza e ultima applicazione di Intelligenza Artificiale relativa a questa tesi, ovvero la sentiment analysis. Inizieremo con un'introduzione all'argomento e poi approfondiremo gli aspetti più tecnici, come il funzionamento e le varie tipologie. Successivamente, esamineremo i casi d'uso più comuni e affronteremo le sfide associate alla sentiment analysis. Chiuderemo il capitolo con la descrizione dell'ultimo case study, che include un'analisi dell'architettura e delle tecnologie impiegate, seguita dalla valutazione dell'applicazione.

5.1 Introduzione alla sentiment analysis

La sentiment analysis, conosciuta anche come opinion mining, è un metodo nell'ambito dell'elaborazione del linguaggio naturale (Natural Language Processing - NLP) che coinvolge l'uso di tecnologie come data mining[1], machine learning, Intelligenza Artificiale e linguistica computazionale per estrarre e valutare il sentimento o l'opinione espressa all'interno di un testo. Questo testo può essere costituito da una vasta gamma di fonti, come recensioni online, post sui social media, articoli di giornale, e-mail, sondaggi e altro ancora.

Attraverso la sentiment analysis, le aziende possono ottenere insight preziosi dalle conversazioni dei clienti e dai flussi dei social media, andando oltre le metriche di base e comprendendo a fondo i sentimenti dei clienti. Questo approccio consente di migliorare l'esperienza dei clienti e di perfezionare le offerte di prodotti o servizi. Inoltre, la sentiment analysis non solo identifica il sentimento, ma può anche estrarre informazioni sulla polarità, la quantità di positività e negatività, il soggetto e il detentore dell'opinione all'interno del testo, consentendo un'analisi dettagliata a vari livelli di granularità.

Il funzionamento della sentiment analysis è caratterizzato da una fase di pre-elaborazione, in cui si identificano le parole chiave per evidenziare il messaggio centrale del testo. Ciò coinvolge la tokenizzazione per suddividere una frase in token, la lemmatizzazione per convertire le parole nella loro radice, e la rimozione delle stop word[2] per filtrare parole non significative. Le tecnologie NLP analizzano, poi, ulteriormente le parole chiave estratte e assegnano ad esse un punteggio di sentiment.

[1] L'estrazione di dati, o data mining, è l'insieme di tecniche e metodologie che hanno per oggetto l'estrazione di informazioni utili da grandi quantità di dati

[2] Le stop word sono parole comuni che vengono spesso eliminate poiché considerate poco informative e non contribuiscono significativamente alla comprensione del significato del testo. Di solito, sono parole come articoli, preposizioni, congiunzioni, e alcuni verbi ausiliari.

I software di sentiment analysis possono utilizzare tre tipologie di approcci per identificare e valutare il tono emotivo o l'opinione manifestata all'interno di un testo; essi sono:

- *L'approccio basato su regole:* identifica, classifica e assegna un punteggio a parole chiave specifiche in base a lessici predeterminati. Ad esempio, le parole in un lessico positivo potrebbero includere "economico", "veloce" e "ben fatto", mentre le parole in un lessico negativo potrebbero presentare "costoso", "lento" e "mal fatto". Questo metodo è semplice da impostare ma difficile da scalare, poiché richiede espansione continua dei lessici e potrebbe non essere accurato in contesti culturali diversi.
- *L'approccio basato sul machine learning:* utilizza algoritmi di classificazione del sentiment per insegnare al software a identificare il sentiment emotivo dal testo. Durante l'addestramento, i data scientist utilizzano set di dati per addestrare il software a riconoscere il sentiment con elevata precisione. Questo approccio è vantaggioso per la sua capacità di elaborare accuratamente un'ampia gamma di informazioni di testo, ma richiede addestramento specifico per un'area di business.
- *L'approccio ibrido:* combina i due approcci precedenti per ottimizzare l'accuratezza e la velocità. Sebbene estremamente accurato, questo approccio richiede più risorse, come tempo e capacità tecniche, rispetto agli altri due.

Nella Figura 5.1 vengono mostrati due approcci alla sentiment analysis.

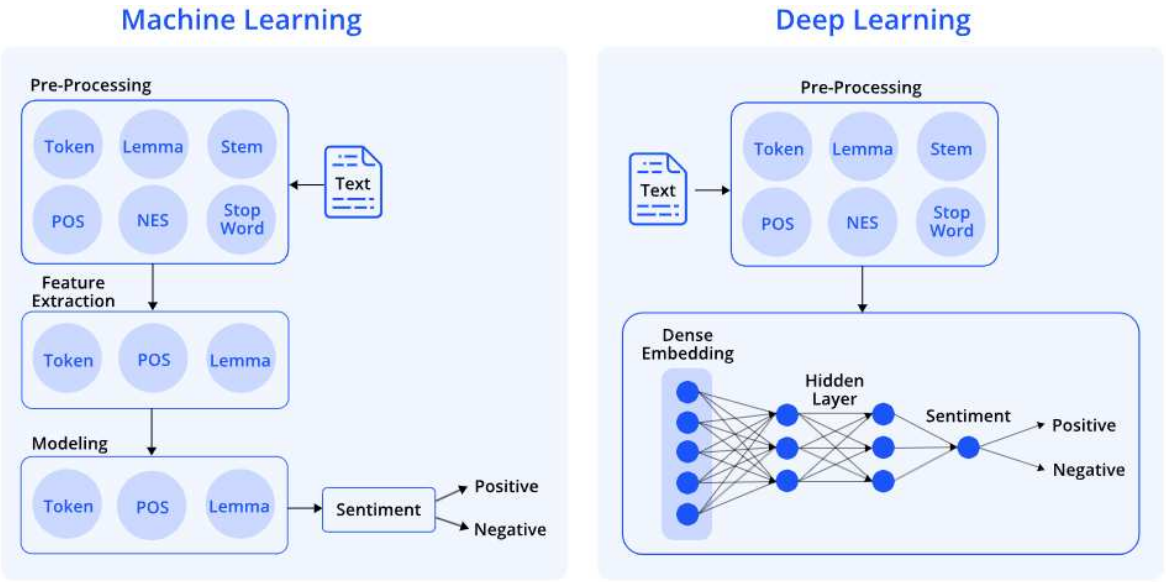


Figura 5.1: ML e DL nella sentiment analysis

5.1.1 Tipologie di sentiment analysis

Oltre ai vari approcci utilizzati per sviluppare gli strumenti di sentiment analysis, esistono anche diversi tipi sentiment analysis che possono essere adottati in base alle proprie esigenze. Tra i quattro più popolari, ovvero basati sugli intenti, basati sull'emozione, a grana fine e basati sugli aspetti (ABSA), tutti sfruttano la capacità sottostante del software di valutare la polarità, ovvero il sentimento complessivo trasmesso da un testo. In generale, la polarità di un testo può essere descritta come positiva, negativa o neutra. Tuttavia, alcuni modelli di sentiment analysis possono suddividere ulteriormente il testo, ad esempio in sottogruppi

come "estremamente positivo" o "estremamente negativo", per individuare emozioni più sfumate e complesse. La polarità viene comunemente espressa come una valutazione numerica su una scala da zero a cento, dove 0 rappresenta un sentimento neutro e 100 rappresenta il sentimento più estremo.

Di seguito sono riportati i quattro tipi di sentiment analysis più diffusi:

- *Analisi a grana fine (Graduata):* questa tecnica raffina la polarità del sentimento distinguendo i gradi di sentimento. Invece di classificare semplicemente i sentimenti come positivi, negativi o neutrali, li suddivide in categorie più specifiche, come "molto positivo", "piuttosto positivo", "neutro", "piuttosto negativo" o "molto negativo". Questo è particolarmente utile per l'analisi dettagliata di recensioni o valutazioni, offrendo una visione granulare dei sentimenti dei clienti.

- *Analisi basata sugli aspetti:* questo approccio ci consente di comprendere il sentimento legato a specifici aspetti di prodotti o servizi. Invece di limitarsi a determinare se il sentimento sia positivo o negativo, scende nei dettagli per scoprire il sentimento riguardante singole caratteristiche o aspetti. Ad esempio, nel caso di un ristorante, gli aspetti potrebbero includere "qualità del cibo", "ambiente" o "velocità del servizio".

- *Rilevamento emotivo:* questa forma di analisi del sentimento identifica emozioni specifiche all'interno dei dati testuali. Invece di determinare se il sentimento sia positivo, neutro o negativo, categorizza le emozioni espresse nel testo, come gioia, sorpresa, rabbia, tristezza, paura, ecc. Ciò offre una comprensione più profonda dello stato emotivo dell'utente.

- *Analisi basata sugli intenti:* questo tipo di analisi mira a comprendere l'intenzione o l'obiettivo dietro un testo specifico. Identificando l'intento sottostante, le organizzazioni possono ottenere approfondimenti sul comportamento del cliente, prevedere azioni future e adattare di conseguenza le proprie strategie. È particolarmente utile in scenari come il servizio clienti, dove prevedere il comportamento di un cliente può aiutare a pianificare risposte efficaci.

5.1.2 L'importanza e le sfide associate alla sentiment analysis

Con l'avvento di una crescente varietà di piattaforme online per l'espressione dei sentimenti personali, le aziende hanno bisogno di strumenti efficaci per monitorare in tempo quasi reale ciò che viene detto su di esse, i loro prodotti e servizi. Progressivamente, sempre più imprese adottano la sentiment analysis e ne fanno uso per esaminare un numero maggiore di conversazioni e interazioni, rendendo più agevole l'individuazione dei punti critici nell'esperienza del cliente.

La sentiment analysis consente di ottenere dati obiettivi, eliminando i pregiudizi personali associati ai revisori umani attraverso l'utilizzo di strumenti basati sull'Intelligenza Artificiale. Questo approccio assicura risultati coerenti e non influenzati durante l'analisi delle opinioni dei clienti. Per esempio, quando un cliente esprime un'opinione negativa insieme a un'opinione positiva in una recensione, un revisore umano potrebbe erroneamente catalogarla come negativa, trascurando le parole positive. Invece, la sentiment analysis potenziata dall'IA permette di valutare in modo obiettivo il testo, riflettendo entrambi i sentimenti.

Le aziende si trovano spesso a gestire enormi quantità di dati non strutturati provenienti da diverse fonti come, e-mail, chatbot, sondaggi e feedback sui prodotti. La sentiment analysis basata sull'IA, con la sua capacità di estrarre rapidamente e in modo efficiente insight significativi da tali dati, rappresenta una soluzione preziosa. Questi strumenti, spesso ospitati

su cloud, consentono alle aziende di scalare la loro capacità di analizzare i dati dei clienti, evitando sprechi di risorse.

Infine, nell'era digitale è essenziale per le aziende rispondere tempestivamente a potenziali crisi o problemi di reputazione sui social media. Gli strumenti avanzati di sentiment analysis potenziati dall'IA consentono agli esecutivi di monitorare il sentiment generale intorno al proprio marchio, individuando rapidamente eventuali segnali di criticità e affrontandoli prontamente per preservare la reputazione aziendale.

La sentiment analysis, pur avanzando rapidamente grazie alle tecnologie di elaborazione del linguaggio naturale (NLP), si trova ancora di fronte a sfide significative. La difficoltà nel comprendere il linguaggio umano si riflette nell'interpretazione errata delle sfumature di comunicazione, come il sarcasmo, la negazione e la multipolarità. L'ironia e il sarcasmo risultano particolarmente problematici per i computer, poiché la loro interpretazione dipende spesso dal tono e dall'espressione facciale. La negazione, ovvero l'uso di parole negative per invertire il significato, può ingannare gli algoritmi, mentre la multipolarità, ovvero la presenza di più sentimenti in una frase, richiede un'analisi più approfondita. Anche il linguaggio idiomatico presenta ostacoli, poiché frasi comuni possono essere fraintese o ignorate dai sistemi di analisi. Nonostante i progressi, resta, quindi, la necessità di perfezionare la sentiment analysis per renderla più accurata e affidabile nel contesto del linguaggio umano.

5.1.3 Applicazioni della sentiment analysis

La sentiment analysis basata sull'IA ha un ruolo cruciale in diversi settori industriali, tra cui la vendita al dettaglio, l'ospitalità, le telecomunicazioni, il settore sanitario e bancario, offrendo una serie di vantaggi significativi. Nei negozi al dettaglio, aiuta i rivenditori a comprendere meglio le opinioni dei clienti, a personalizzare il marketing, a migliorare il servizio clienti, a sviluppare prodotti e ad anticipare le tendenze di mercato. Nell'ospitalità, consente di valutare i feedback degli ospiti da varie fonti online, migliorare l'esperienza complessiva degli ospiti, prendere decisioni strategiche e individuare tendenze emergenti. Nel settore delle telecomunicazioni, guida il miglioramento dell'esperienza del cliente, la gestione del rischio, lo sviluppo di prodotti e servizi, l'analisi della concorrenza e la reattività in tempo reale alle questioni emergenti. Nel settore sanitario, migliora l'assistenza ai pazienti, la gestione dei servizi e la sicurezza dei farmaci. Infine, nel settore bancario, ottimizza l'esperienza del cliente, la gestione del rischio, la percezione del marchio, il marketing e le vendite, nonché il rilevamento delle frodi, contribuendo a migliorare l'efficienza operativa e la competitività complessiva delle istituzioni finanziarie.

5.2 Descrizione del case study

In questa sezione esamineremo in dettaglio il terzo case study sviluppato che, con l'uso di Amazon Comprehend, si focalizza sull'implementazione di un'applicazione che consente di analizzare le recensioni o i feedback dei clienti o degli utenti di un servizio. Gli utenti avranno la possibilità di esprimere la propria opinione sia tramite testo che attraverso una valutazione a stelle. In risposta, riceveranno una stima del punteggio in stelle derivato dal testo e un grafico che illustra le principali entità e i sentiment associati ad esse.

5.2.1 Architettura dell'applicazione

Inizieremo dando un'overview sui vari servizi e tecnologie utilizzati, lasciando i dettagli implementativi per le sezioni successive. Nella Figura 5.2 viene presentata l'architettura dei servizi AWS impiegati per sviluppare l'applicazione.

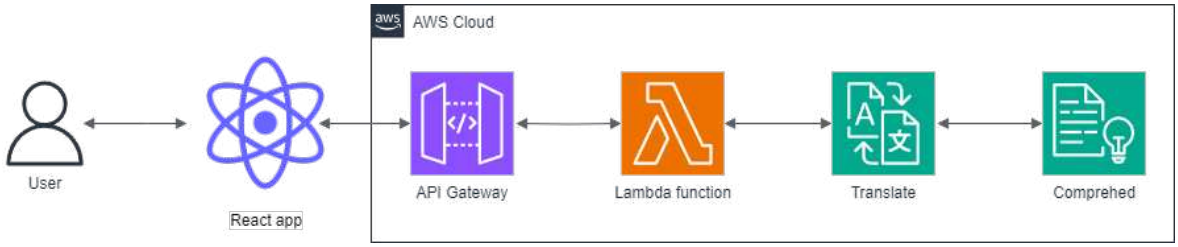


Figura 5.2: Architettura dell'applicazione di sentiment analysis

Utilizzando l'interfaccia grafica, l'utente potrà selezionare il numero di stelle che desidera assegnare al servizio e fornire una recensione in forma di testo. Tali informazioni saranno elaborate da una funzione Lambda attivata da una REST API che fornisce un endpoint alla suddetta funzione. Lo scopo della funzione Lambda sarà quello di calcolare il punteggio in stelle associato al testo e generare i dati necessari per la visualizzazione del grafico.

5.2.2 Tecnologie di front-end

Nello sviluppo del front-end dell'applicazione sono state utilizzate le stesse tecnologie viste nel capitolo precedente, ovvero la libreria React JS.

Nella Figura 5.3 viene mostrata l'interfaccia grafica realizzata. Nell'immagine non è presente il grafico poiché viene generato dinamicamente insieme ai dati ottenuti dalle elaborazioni. Sarà osservabile durante la fase di testing.

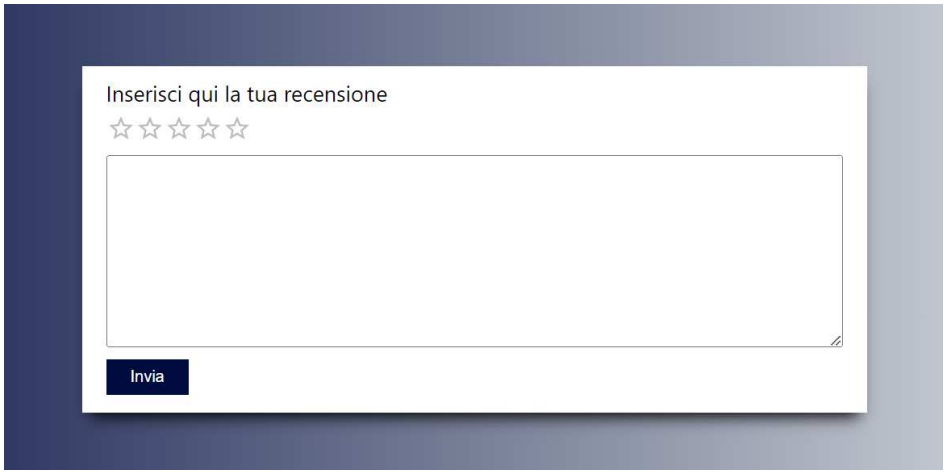


Figura 5.3: Interfaccia grafica dell'applicazione di sentiment analysis

Continuiamo, ora, focalizzandoci sullo studio delle due funzioni del codice React, che si occupano di inviare richieste HTTP al back-end e di ricevere le relative risposte. Nella Figura 5.4 viene illustrata la funzione principale, mentre nella Figura 5.5 viene mostrata la funzione ausiliaria.

Iniziamo con la funzione `getResponse`, poiché ci aiuterà a comprendere la struttura della funzione `sendReview`. Essa invia una richiesta HTTP, contenente il testo della recensione come parametro URL, all'endpoint dell'API. Tale richiesta utilizza il metodo `GET`, poiché stiamo recuperando dati dal server, e utilizza un header per specificare il formato dei dati inviati e ricevuti. Infine, la risposta viene convertita in formato JSON utilizzando il metodo `then` e restituita alla funzione chiamante.

La funzione `sendReview` viene attivata quando l'utente clicca il pulsante "Invia" ed effettua un controllo per verificare l'inserimento dei dati. In caso negativo, la funzione termina.

```
const sendReview = async () => {
    if (reviewText && reviewStar) {
      const res = await getResponse();
      if (res) {
        const resArray = Object.values(res);
        setReviewData(resArray[1]);

        if (typeof (resArray[0]) === 'string') {
          setInfo(resArray[0]);
          setReviewStarFromAWS(3);
        } else {
          setInfo('La recensione corrisponde ad un rating di');
          setReviewStarFromAWS(resArray[0]);
          setDisplayChart(true);
        }
      }
    }
  }
```

Figura 5.4: Codice della funzione `sendReview`

```
const getResponse = async () => {
    const requestURL = 'https://c5wk4pir8i.execute-api.eu-west-2.amazonaws.com/prod/review_data?'
    + new URLSearchParams({ reviewText: reviewText });
    return await fetch(requestURL, {
      method: 'GET',
      headers: {
        'Accept': 'application/json',
        'Content-Type': 'application/json',
      }
    })
      .then(response => response.json())
      .then((data) => {
        return data;
      })
      .catch(error => console.error(error))
}
```

Figura 5.5: Codice della funzione `getResponse`

Nell'altro caso viene invocata la funzione `getResponse` e si attende di ricevere una risposta che, successivamente, viene convertita in una struttura dati di tipo `Array`. La nuova risposta viene utilizzata per configurare le informazioni da mostrare all'utente; in particolare, viene utilizzata per effettuare un controllo sul tipo, poiché, ci sono situazioni in cui la sentiment analysis restituisce un risultato neutro; in tal caso, viene assegnato automaticamente un punteggio di tre stelle, invece; in tutti gli altri casi, è Amazon Comprehend a stabilirlo.

5.2.3 Tecnologie di back-end e API

Come già introdotto in precedenza, la gestione dei dati lato server è stata effettuata mediante una funzione Lambda che viene attivata quando arriva una richiesta HTTP all'endpoint dell'API. Nella Figura 5.6 è possibile osservare la struttura dell'API e la configurazione del trigger.

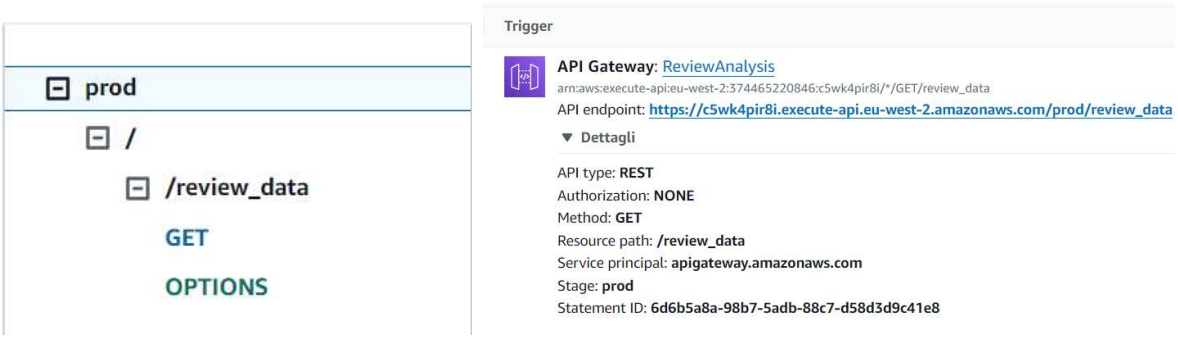


Figura 5.6: Configurazione dell'API `ReviewAnalysis` e del trigger per la funzione Lambda

Adesso, procediamo ad esaminare attentamente la funzione Lambda per comprendere il processo di elaborazione dei dati al fine di ottenere le risposte desiderate. Iniziamo mostrando, in Figura 5.7, la funzione principale `lambda_handler` e, successivamente, le due funzioni ausiliarie.

```python
def lambda_handler(event, context):

    reviewText = event['queryStringParameters']['reviewText']

    Translate_client = boto3.client('translate')
    Comprehend_client = boto3.client('comprehend')

    responseTranslate = Translate_client.translate_text(
        Text=reviewText,
        SourceLanguageCode='auto',
        TargetLanguageCode='en',
    )

    responseComprehend = Comprehend_client.detect_targeted_sentiment(
        Text=responseTranslate["TranslatedText"],
        LanguageCode='en'
    )

    rating = get_rating(responseComprehend)
    chart_data = get_chart_data(responseComprehend)

    response = {
        'rating': rating,
        'chart_data': chart_data
    }

    return build_response(200, response)
```

Figura 5.7: Codice della funzione `lambda_handler` di `ReviewAnalysis`

Quando attivata tramite una richiesta HTTP, la funzione estrae il parametro `review_data` dall'URL e, successivamente, stabilisce le connessioni ai servizi AWS. È stato impiegato Amazon Translate per convertire con precisione il testo della recensione poiché, secondo la documentazione ufficiale di Amazon Comprehend, l'API `detect_targeted_sentiment` supporta esclusivamente la lingua inglese. Con Amazon Comprehend, invece, è stato analizzato il testo della recensione ed è stato ottenuto un file JSON contenente tutte le entità e i relativi sentiment. Successivamente, questo file JSON viene passato come argomento alle due funzioni ausiliarie, le quali restituiranno i dati da mostrare all'utente.

Mediante la funzione `get_rating`, mostrata in Figura 5.8, viene calcolato il numero di stelle corrispondente al sentiment generale della recensione, utilizzando i dati presenti nel file JSON `responseComprehend`.

```python
def get_rating(responseComprehend):

    counterPositive = 0
    counterNegative = 0
    counterNeutral = 0
    numberOfSentimentWithNeutral = 0

    for entity in responseComprehend["Entities"]:
        if entity["Mentions"][0]["MentionSentiment"]["Sentiment"] == "POSITIVE":
            counterPositive += 5
        elif entity["Mentions"][0]["MentionSentiment"]["Sentiment"] == "NEGATIVE":
            counterNegative += 1
        else:
            if entity["Mentions"][0]["MentionSentiment"]["SentimentScore"]["Neutral"] == 1.0:
                numberOfSentimentWithNeutral += 1
            elif entity["Mentions"][0]["MentionSentiment"]["SentimentScore"]['Positive']
            > entity["Mentions"][0]["MentionSentiment"]["SentimentScore"]['Negative']:
                counterNeutral += 5
            else:
                counterNeutral += 1

    numberOfSentiment = len(
        list(responseComprehend["Entities"])) - numberOfSentimentWithNeutral
    if (numberOfSentiment == 0):
        return "L'analisi ha dato come risultato un sentimento totalmente neutro"

    rating = (counterPositive+counterNegative+counterNeutral)/numberOfSentiment

    return int(rating)
```

Figura 5.8: Codice della funzione `get_rating` di `ReviewAnalysis`

La logica della funzione si fonda sul calcolo del numero di stelle medio in base ai sentiment delle entità. In particolare, ad ogni entità con sentiment totalmente positivo o negativo, viene associato un punteggio, rispettivamente, di cinque stelle o di una stella. Se il sentiment per un'entità è, invece, misto, viene valutato il `SentimentScore`, una struttura dati di tipo `dizionario` che contiene la percentuale di ciascun sentiment associato a un'entità. Se il `SentimentScore` è neutro al 100%, quell'entità non viene inclusa nel punteggio; invece, se prevale la percentuale di sentiment positivo o negativo, viene associato il punteggio di cinque stelle o una stella, rispettivamente.

Dopo aver valutato tutte le entità, si procede calcolando il numero totale di sentiment, escludendo quelli neutri. Se il risultato è zero, viene restituito all'utente un messaggio e

assegnato automaticamente il punteggio di tre stelle, che rappresenta un sentiment neutro; altrimenti, viene calcolata la media dei sentiment riscontrati per determinare il numero di stelle, che viene, quindi, restituito all'utente.

La funzione finale, denominata `get_chart_data`, è incaricata di preparare i dati necessari per il grafico che verrà mostrato all'utente. Descritta nella Figura 5.9, essa utilizza una struttura dati complessa in cui il primo elemento è un'array contenente le etichette da mostrare sull'asse delle ascisse, mentre gli altri elementi sono array che includono il nome dell'entità e i relativi valori percentuali per ciascun sentiment. Questa struttura dati viene, poi, restituita all'utente per consentire la visualizzazione del grafico.

```python
def get_chart_data(responseComprehend):

    chart_data = []
    sentiment = ['Entity', 'Positive', 'Negative', 'Mixed', 'Neutral']
    chart_data.append(sentiment)

    for entity in responseComprehend["Entities"]:
        data = []
        data.append(entity["Mentions"][0]["Text"])
        data.append(entity["Mentions"][0]["MentionSentiment"]
                    ["SentimentScore"]["Positive"]*100)
        data.append(entity["Mentions"][0]["MentionSentiment"]
                    ["SentimentScore"]["Negative"]*100)
        data.append(entity["Mentions"][0]["MentionSentiment"]
                    ["SentimentScore"]["Mixed"]*100)
        data.append(entity["Mentions"][0]["MentionSentiment"]
                    ["SentimentScore"]["Neutral"]*100)
        chart_data.append(data)

    return chart_data
```

Figura 5.9: Codice della funzione `get_chart_data` di `ReviewAnalysis`

5.2.4 Test dell'applicazione

Per illustrare il funzionamento dell'applicazione verranno utilizzate delle recensioni prese da alcuni siti online affinché il risultato sia il più verosimile possibile.

La prima recensione a cui sono state assegnate due stelle riguarda un ristorante e si presenta nel seguente modo:

> *Ieri per un compleanno siamo andati in questo ristorante, pizzeria. Non mi è piaciuto perché l'ambiente era scarno e di pizzeria non aveva nulla. La pizza non era buona. Unica nota positiva la cordialità e la gentilezza del personale.*

La seconda recensione, da quattro stelle, proviene da un hotel ed è così formulata:

Figura 5.10: Risultati del primo test

Buon hotel, pulito, adiacente all'omonimo ippodromo. Noi abbiamo soggiornato nella suite, molto spaziosa, pulita con bagno molto grande. Colazione nella norma, da migliorare aggiungendo altre scelte. Unica nota negativa: ci hanno chiamato la mattina per spingerci a fare un check out in anticipo rispetto all'ora ufficiale.

Figura 5.11: Risultati del secondo test

In questo caso l'analisi della recensione ha portato ad un risultato diverso da quello espresso dall'utente; secondo l'analisi vi sono stati dei sentiment molto negativi nella parte finale della recensione che hanno portato ad abbassare di una stella il punteggio totale.

Per effettuare un ultimo test utilizziamo una recensione con un testo più lungo in cui vi sono vari sentiment contrapposti; la recensione che segue è ancora di un ristorante ed è stata valutata con tre stelle:

> *Buona scelta di pizze regionali e qualche altro piatto che non ho assaggiato. Pizza soffice e digeribile, discreti ingredienti. Purtroppo il problema di questo locale è il servizio: camerieri poco attenti e distratti che vagano come anime in pena mancando evidentemente una direzione di sala adeguata. Attese piuttosto lunghe soprattutto nelle ore di punta dove si ha la pretesa di servire contemporaneamente in sala e preparare le pizze da asporto. Abbastanza rumoroso all'interno. Molto conveniente il menù del giorno.*

Figura 5.12: Risultati del terzo test

CAPITOLO 6

Discussione

In questo capitolo verranno esposte alcune riflessioni e considerazioni in merito al lavoro svolto e verranno tratte delle conclusioni chiave che metteranno in luce le principali scoperte emerse da ciascuna delle esperienze trattate nei capitoli precedenti.

6.1 Riflessioni e considerazioni generali

Il capitolo conclusivo è un momento cruciale per riflettere sulle tre esperienze significative condotte nell'ambito dei chatbot, del riconoscimento delle immagini e della sentiment analysis. Attraverso queste esperienze, sono stati esplorati e analizzati diversi aspetti legati all'Intelligenza Artificiale e al suo impatto su vari contesti applicativi. Le sfide affrontate, le metodologie utilizzate e i risultati ottenuti forniscono una panoramica completa delle potenzialità e dei limiti di tali tecnologie.

Durante il percorso di esplorazione delle tre esperienze nell'ambito dell'Intelligenza Artificiale ho avuto l'opportunità di cogliere una serie di aspetti positivi e negativi che delineano la complessità di queste tecnologie. Tra gli aspetti positivi, spicca sicuramente la potenza trasformativa di queste applicazioni nel migliorare l'efficienza e l'accessibilità dei servizi digitali, facilitando l'interazione uomo-macchina in modo più intuitivo e naturale. Inoltre, ho potuto apprezzare la vastità delle possibilità offerte nella risoluzione di problemi complessi, come il riconoscimento di immagini e la sentiment analysis. Tuttavia, è emerso anche il lato critico di queste tecnologie, evidenziando sfide legate alla privacy, alla sicurezza dei dati e alla possibile creazione di bias algoritmici. Nel complesso, queste esperienze hanno arricchito notevolmente il mio bagaglio culturale e conoscitivo, consentendomi di comprendere in maniera più approfondita il ruolo dell'Intelligenza Artificiale nella società contemporanea. Esse mi hanno anche spinto a riflettere criticamente sulle implicazioni etiche e sociali di queste tecnologie, alimentando il mio interesse per la ricerca e l'innovazione in questo campo in continua evoluzione.

6.2 Discussione dell'esperienza sui chatbot

Lo sviluppo di un chatbot mi ha permesso di esplorare le potenzialità e le sfide dell'Intelligenza Artificiale nell'ambito delle interazioni conversazionali. Durante lo sviluppo, ho potuto apprezzare l'opportunità offerta dal chatbot nel migliorare l'accessibilità e l'efficienza dei servizi, fornendo risposte immediate e personalizzate agli utenti. Tuttavia, ho anche

incontrato diverse difficoltà nell'addestramento del modello per comprendere e rispondere in modo accurato alle domande degli utenti e nell'assicurare una gestione coerente e soddisfacente delle conversazioni. Il servizio Amazon Lex di AWS ha certamente offerto vantaggi significativi nel processo di sviluppo del chatbot. La sua console intuitiva e le ampie funzionalità hanno reso il lavoro più agevole. Inoltre, l'integrazione con altri servizi di AWS ha fornito un ecosistema completo per gestire e scalare il chatbot in base alle esigenze. Tuttavia, è importante considerare attentamente anche i limiti associati all'uso di Amazon Lex, come la necessità di garantire la conformità alle normative sulla privacy e sulla sicurezza dei dati. In aggiunta, sebbene possa sembrare di poco conto, Amazon Lex permette di distribuire il chatbot solo in alcune applicazioni di messaggistica; ad esempio, per la distribuzione su Telegram è stato necessario creare un'API apposita.

6.3 Discussione dell'esperienza sul riconoscimento immagini

Durante il percorso di esplorazione nel campo del riconoscimento delle immagini, ho avuto l'opportunità di sviluppare un'applicazione che permette agli utenti di effettuare l'accesso mediante il riconoscimento facciale. Attraverso l'utilizzo di strumenti avanzati come Amazon Rekognition, basati su tecniche di deep learning, ho potuto esplorare le possibilità offerte da questa tecnologia avanzata e integrarle in un'applicazione pratica e funzionale. Tra i punti di forza di Amazon Rekognition si evidenzia, innanzitutto, la sua alta precisione nel riconoscimento facciale, che consente un'identificazione rapida e affidabile degli utenti. I modelli di deep learning utilizzati per il riconoscimento facciale sono in grado di apprendere e identificare pattern complessi nelle immagini, garantendo una elevata accuratezza nelle identificazioni. Inoltre, la facilità d'uso e l'integrazione semplice con altre piattaforme e servizi AWS hanno semplificato notevolmente il processo di sviluppo e implementazione dell'applicazione.

Tuttavia, nonostante i suoi indiscutibili vantaggi, il riconoscimento facciale presenta anche alcuni limiti e sfide da considerare attentamente. Uno dei principali risiede nelle preoccupazioni sulla privacy e sulla sicurezza dei dati biometrici degli utenti. È essenziale garantire che le informazioni facciali degli utenti siano gestite e protette in conformità con le normative sulla privacy e le migliori pratiche di sicurezza. Inoltre, è importante tenere in considerazione che il riconoscimento facciale potrebbe non essere sempre accurato al 100%, con il rischio di falsi positivi o falsi negativi che potrebbero compromettere l'accesso degli utenti o consentire l'accesso non autorizzato. Pertanto, è fondamentale valutare attentamente l'efficacia e l'attendibilità della tecnologia e adottare misure appropriate per mitigare i rischi associati al suo utilizzo.

6.4 Discussione dell'esperienza sulla sentiment analysis

Durante l'esperienza di sentiment analysis, ho avuto l'opportunità di esplorare il potenziale dell'Intelligenza Artificiale nel comprendere e analizzare le emozioni umane attraverso testo scritto. Con l'utilizzo di strumenti avanzati come Amazon Comprehend, basati su tecnologie di machine learning e NLP (Natural Language Processing), ho potuto utilizzare la sentiment analysis per comprendere e valutare le emozioni espresse nelle recensioni e nei commenti online. Tra i punti di forza di questa tecnologia spicca, innanzitutto, la sua capacità di analizzare grandi volumi di testo in modo rapido ed efficiente, consentendo di estrarre insight significativi e utili per prendere decisioni informate. Inoltre, la precisione e l'accuratezza degli algoritmi di Amazon Comprehend hanno reso possibile identificare e

categorizzare con precisione i sentimenti espressi nei testi, fornendo una panoramica chiara e dettagliata delle opinioni degli utenti.

Tuttavia, è importante considerare anche alcuni limiti di questa tecnologia. Ad esempio, la sentiment analysis potrebbe non essere sempre in grado di catturare la complessità e il contesto delle emozioni umane, rischiando di interpretare erroneamente il tono di un testo. Inoltre, la presenza di linguaggio figurativo, sarcasmo o ironia potrebbe complicare ulteriormente l'accuratezza dell'analisi. Nonostante questi limiti, l'esperienza di sviluppo di un'applicazione per la sentiment analysis ha evidenziato il potenziale significativo di questa tecnologia nel comprendere le opinioni degli utenti e nel prendere decisioni informate basate sulle loro emozioni.

Conclusioni

Nel corso di questa tesi abbiamo condotto e analizzato tre case study riguardanti diverse applicazioni dell'Intelligenza Artificiale, facendo ampio uso dei servizi offerti da AWS. Le esperienze affrontate hanno spaziato su tematiche ampiamente diffuse, quali i chatbot, il riconoscimento facciale e l'analisi dei sentimenti.

Nel case study dedicato al chatbot abbiamo inizialmente selezionato un contesto applicativo rilevante nel settore del fitness. Successivamente, abbiamo realizzato il *conversation flow diagram* e abbiamo configurato gli intent tramite la console di Amazon Lex. Infine, abbiamo distribuito il chatbot completo sulla piattaforma Telegram, sfruttando servizi come Amazon API Gateway e Amazon DynamoDB.

Il secondo case study si è concentrato sullo sviluppo di un'applicazione di riconoscimento facciale. Qui, abbiamo realizzato un'architettura complessa, coinvolgendo vari servizi AWS e implementando un'applicazione front-end con la libreria React JS. Dopo la progettazione dell'interfaccia grafica, abbiamo affrontato l'organizzazione del back-end e dell'API, fino alla definizione dello storage dei dati. Infine, abbiamo implementato e testato le API attraverso Amazon API Gateway.

Nell'ultimo case study abbiamo esplorato le possibilità offerte dalla sentiment analysis. L'obiettivo principale è stato quello di valutare la coerenza tra il testo di una recensione e il numero di stelle assegnate, sfruttando i dati ottenuti dall'analisi testuale tramite IA. Anche in questa occasione abbiamo seguito un approccio simile al caso precedente, partendo dalla progettazione e implementazione dell'interfaccia grafica, fino alla realizzazione e al test delle API.

Gli esperti guardano con ottimismo al progresso futuro dell'IA, con stime che prevedono un notevole contributo all'economia nei prossimi dieci anni. Tuttavia, non mancano le voci critiche che sollevano dubbi sull'arrivo dell'IA generale e evidenziano preoccupazioni etiche sull'equità e sulla sicurezza. Il progresso dell'IA finora è stato guidato da competizioni volte a creare soluzioni migliori per affrontare problemi specifici o compiti ben definiti, ma per raggiungere l'IA generale sono necessari nuovi approcci e progressi tecnologici. Jeff Dean di Google propone un approccio rivoluzionario, immaginando un'unica vasta Intelligenza Artificiale in grado di adattarsi a una molteplicità di compiti. Questo richiede un'innovazione continua sia nelle componenti che nelle architetture dei sistemi di IA. L'ingegneria dell'IA deve essere in grado di supportare l'implementazione su larga scala di sistemi complessi, integrando nuove idee e tecnologie. In definitiva, il futuro dell'IA si basa sull'innovazione continua e sull'esplorazione di nuove frontiere, come dimostrano sviluppi recenti quali i modelli linguistici trasformativi, come BERT e GPT-4, e le reti neurali, che scalano fino a 68 miliardi di parametri. In conclusione, l'IA ha compiuto grandi progressi nella sua storia, ma

una frase di Alan Turing, riportata nel suo saggio del 1950, è ancora valida oggi:

> *Possiamo vedere solo una breve distanza davanti a noi, ma possiamo vedere che molto resta da fare.*

Bibliografia

AARONSON, S. (2014), *My conversation with "Eugene Goostman," the chatbot that's all over the news for allegedly passing the Turing test*, Shtetl-Optimized.

BURKOV, A. (2019), *The Hundred-Page Machine Learning Book*, Andriy Burkov.

GENESERETH, M. R. e NILSSON, N. J. (1987), *Logical Foundations of Artificial Intelligence.*, Morgan Kaufmann.

JEFF HAWKINS, S. B. (2004), *On Intelligence: How a New Understanding of the Brain Will Lead to the Creation of Truly Intelligent Machines*, Times Books.

MIIKKULAINEN, M., LIANG (2019), *Evolving deep neural networks. In Artificial Intelligence in the Age of Neural Networks and Brain Computing.*, Elsevier.

MINSKY, M. L. (2007), *The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind.*, Simon and Schuster.

NILSSON, N. J. (1971), *Problem-Solving Methods in Artificial Intelligence.*, McGraw-Hill.

RUSSEL, S. (2020), *Human Compatible: AI and the Problem of Control*, Penguin.

STUART RUSSEL, P. N. (2021), *Artificial Intelligence A Modern Approach.*, Pearson.

TEGMARK, M. (2018), *Life 3.0: Being Human in the Age of Artificial Intelligence*, Penguin.

WILKS, Y. (2019), *Artificial Intelligence: Modern Magic or Dangerous Future.*, Icon.

YAMPOLSKIY, R. V. (2018), *Artificial Intelligence Safety and Security.*, Chapman and Hall/CRC.

Siti web consultati

- Amazon Web Service – `https://aws.amazon.com/`
- Red Hat – `https://www.redhat.com/it`
- IBM – `https://www.ibm.com/it-it`

- Blog Google – `https://blog.google/intl/it-it/`
- Oracle – `https://www.oracle.com/it/cloud/`
- Intelligenza Artificiale Italia – `https://www.intelligenzaartificialeitalia.net/`
- Nanonets – `https://nanonets.com/blog/image-recognition/`
- My Great Learning – `https://www.mygreatlearning.com/blog/image-recognition`
- LeewayHertz – `https://www.leewayhertz.com/`
- Precedence Research – `https://www.precedenceresearch.com/`
- JavaTPoint – `https://www.javatpoint.com`
- Digital4 – `https://www.digital4.biz`

Ringraziamenti

Innanzitutto vorrei ringraziare il Professor Domenico Ursino per avermi guidato attentamente in questo percorso, mostrando estrema disponibilità e professionalità.

Ringrazio in modo unico Tatiana che è sempre stata al mio fianco a supportarmi e sopportami nei momenti difficili. Mi ha accompagnato in ogni occasione, mostrando sempre il suo interesse e il suo appoggio. Ti ringrazio per questo, ti voglio bene.

Ovviamente, un ringraziamento speciale va ai miei genitori, che hanno sempre creduto in me e mi hanno sostenuto in tutti i modi in questi tre anni.

Non possono mancare i ringraziamenti a Natasha, Igor, Caterina e Alessandro, la mia seconda famiglia, che sono sempre stati presenti a mostrare il loro supporto.

Ci tengo inoltre a ringraziare i miei compagni di corso per avermi fatto passare momenti di svago e momenti di confronto in cui ho potuto imparare cose nuove.

Infine, vorrei ringraziare me stesso, che, con costanza e dedizione, ho sempre portato a termine gli obiettivi che mi ero prefissato.