UNIVERSITÀ DEGLI STUDI DI GENOVA
SCUOLA POLITECNICA

Corso di Laurea in Ingegneria Elettronica
e Tecnologie dell'informazione

Tesi di Laurea

Riconoscimento di significati dipendenti dal contesto nell'interazione uomo-robot

Context-dependent meanings recognition in human-robot interaction

Relatori:
Prof. Antonio Sgorbissa
Dott. Roberto Menicatti

Candidato:
Davide Lanza
Matricola: 4228930

Anno accademico 2017-2018

Sommario

Il riconoscimento di significati dipendenti dal contesto in enunciati del linguaggio naturale è una delle principali problematiche della pragmatica computazionale. Un procedimento inferenziale in particolare, quello abduttivo, sembra essere adatto a modellare e comprendere fenomeni di questo tipo. Nell'abduzione infatti, mediante delle ipotesi, si tentano di spiegare le osservazioni fatte, e questo meccanismo permette di capire come, in mancanza di deduzioni esatte, la comprensione di significati implicati o sottintesi sia comunque possibile.

L'obiettivo di questo studio è analizzare una possibile implementazione di tale procedimento in un sistema robotico interagente con un utente umano. Il sistema in questione quindi dovrà essere in grado di estrarre mediante informazioni contestuali i significati implicati dal parlante. L'approccio convenzionale a queste forme linguistiche (in opposizione all'approccio inferenziale radicale) sostiene che i significati implicati dal parlante siano riconoscibili in quanto codificati in un sistema di convenzioni condivise dall'ascoltatore. A tal proposito, ciò su cui si interroga il lavoro di ricerca qui proposto è l'effettiva accuratezza ed efficacia con cui tale sistema, modellato secondo un approccio convenzionale, possa reagire correttamente a questi fenomeni linguistici.

Per analizzare il problema si è definito un modello di studio ristretto sul quale eseguire un'analisi di efficacia, tralasciando alcune problematiche affrontate successivamente. I risultati ottenuti confermano la validità dell'approccio convenzionale e, su questa base, sono stati proposti alcuni possibili sviluppi ulteriori allo scopo di superare le limitazioni del modello utilizzato.

Ringraziamenti

I miei ringraziamenti vanno a tutte le persone che mi hanno aiutato ed assistito durante la preparazione, la realizzazione e la stesura del seguente elaborato con osservazioni, suggerimenti e critiche.

Ringrazio il Professor Antonio Sgorbissa per avermi permesso di realizzare il presente lavoro di ricerca, con grande disponibilità e cordialità, accettando di essere il mio relatore. Ringrazio il Dottor Roberto Menicatti per il grande supporto durante le fasi di ricerca, di realizzazione e di organizzazione del materiale.

Ringrazio inoltre il Professor Marcello Frixione per il suo aiuto ed i suoi consigli, in particolare per quanto riguarda l'inquadramento concettuale del progetto e gli studi di pragmatica in merito a quanto affrontato.

Un ringraziamento va anche a tutti coloro con cui ho avuto modo di confrontarmi durante le fasi realizzative ed a quelli che hanno gentilmente accettato di leggere e discutere con me il presente elaborato.

Indice

Elenco delle tabelle

Elenco delle figure

Capitolo 1

Introduzione

1.1 Il contesto nelle interazioni uomo-robot

Nell'interazione linguistica tra esseri umani il contesto svolge un ruolo determinante nell'attribuzione dei significati. Quando si sviluppano dei sistemi intelligenti, o sistemi che agiscano *come se* fossero intelligenti, l'attribuzione di significati deve essere modellata in maniera coerente, in modo da fornire risultati efficaci e reazioni simili a quelle di un agente dotato di intelligenza. Per quanto riguarda l'interazione uomo-robot, siamo soliti pensare quest'ultimo come uno strumento in grado di eseguire comandi (od il cui "scopo" sia quello di eseguire comandi). Quando un utente interagisce con un sistema robotico, si aspetta che la parte di sistema dedicata alle funzioni "cognitive", il *software* implementato all'interno del dispositivo meccatronico, comprenda i comandi che l'utente gli rivolge e li trasformi in direttive all'*hardware*, da eseguire in risposta alla richiesta.

Il contesto assume qui un ruolo fondamentale, essendo il tipo di interazione trattato modellato a partire da quanto normalmente avviene negli scambi linguistici tra esseri umani. Se infatti è vero che un utente umano si rapporta ad un sistema robotico con modalità diverse da quelle con cui si rapporta con un essere umano, non tutti i comandi sono espressioni esecutive determinate (quali potrebbero essere "accendimi la luce" oppure "portami una penna"). Molte volte infatti i comandi sono espressi tramite perifrasi, allusioni o manifestazioni di bisogni specifici. Un sistema robotico che sappia agire come un essere dotato di intelligenza deve saper reagire correttamente a questo tipo di stimoli, deve essere in grado di estrarre da queste espressioni un significato ulteriore, diverso da quello letterale, che il parlante veicola

indirettamente mediante allusioni ed implicazioni. La conoscenza del contesto in cui tali enunciati vengono proferiti è, come vedremo, necessario alla comprensione dei significati non letterali (la pragmatica ha studiato ampiamente la fenomenologia di queste interazioni linguistiche) ed un sistema in grado di comprendere come identificare, a partire dal contesto, determinati significati allusivi è quindi un sistema in grado di comprendere ed eseguire comandi velati, un sistema in grado di reagire alle situazioni adeguatamente rispetto al contesto in cui si trova.

In questo elaborato verrà affrontata questa problematica, provando a implementare un sistema in grado di reagire correttamente a stimoli di natura linguistica dipendenti dal contesto. Si è scelto in questa sede di tralasciare la problematica *hardware* legata allo sviluppo del robot in sé, concentrandosi sulla progettazione del sistema di identificazione di significati e reazioni. A partire da una stringa testuale in linguaggio naturale, quindi, il *software* qui proposto dovrà identificare la reazione adeguata al contesto di proferimento, in modo da eseguire comandi non espliciti nella maniera più efficace possibile. Nel seguito del Capitolo si affronterà la problematica dal punto di vista teorico, illustrando gli studi di linguistica svolti in tal senso e come questi possano aiutare a delineare il problema alla base. Nel Capitolo 2 verranno quindi presentati il tipo di problema affrontato nello specifico, la proposta di modellizzazione del contesto adottata e la struttura del sistema che dovrà eseguire riconoscimento e restituzione della reazione adeguata. I Capitoli successivi descriveranno specificamente le fasi di sviluppo, valutazione ed implementazione del sistema.

1.2 Implicazioni ed implicature in pragmatica

La pragmatica è quel settore della linguistica che studia il tipo di relazione che intercorre tra i fenomeni linguistici e gli aspetti contestuali dell'uso di un linguaggio. Si occupa quindi di tutti quei proferimenti (*utterances*) il cui significato non può esaurirsi in ciò che è contenuto strettamente nell'enunciato. Una tipologia importante di proferimenti sono gli indicali (*indexical*) [3], espressioni utilizzate per riferirsi ad un determinato contesto. Mediante gli indicali ci si può riferire a luoghi, persone, istanti od oggetti in quanto in una determinata relazione con il proferente. Contrariamente ad enunciati *context-free*, il cui significato non è dipendente dal contesto (ad esempio "2+2=4", o "I cani sono mammiferi"), i proferimenti indicali sono difficili da comprendere senza avere informazioni riguardo al contesto ed alle relazioni

che il proferente intrattiene con essi (ad esempio "Stanno suonando alla porta", o "Mi sono dimenticato di fare le valigie"). Non solo gli indicali, ma molte tipologie di proferimento non prescindono dal contesto, come ad esempio le espressioni che contengono presupposizioni [6] o implicature conversazionali di un determinato tipo [7] oppure gli atti linguistici indiretti [1]. In questi casi è evidente l'importanza del meccanismo di inferenza durante il processo di comprensione del linguaggio naturale. La condivisione, tra proferente ed ascoltatore, di un certo tipo di conoscenza comune del contesto di proferimento è di peculiare importanza, come evidenziato da [3]:

> *The main characteristic of the inferences studied has been that they rely on premisses which are not part of the content of the utterance itself. Much of (...) early AI work (...) was concerned with the general background knowledge that speaker and hearer must share so that the one can infer what is implied by the other. (...) The source for this perspective on language understanding is usually traced either to Grice or to Sperber and Wilson* [3, pg. 6]

Paul Grice è noto per aver mostrato come una comunicazione efficace si debba fondare sull'assunto che i due individui cooperino al fine di portare a termine lo scambio linguistico (principio di cooperazione griceano). Questa ipotesi ha poi fornito a Grice le premesse per delineare la nozione di implicatura quale forma di implicazione "inaffidabile"(*unreliable*, cfr.[3, pg.6]) . Sebbene sia stata criticata l'enfasi sulla cooperazione (dai sopracitati Sperber e Wilson) rimane molto forte l'idea che l'implicatura sia un modo per veicolare le intenzioni del parlante, come nell'esempio sempre riportato in [3]:

> *There's a howling gale in here! = Shut the door/window.*

In questo caso lo studio delle implicature è utile per spiegare l'atto linguistico indiretto[1], come il proferimento possa veicolare una richiesta quale quella di chiudere la porta o la finestra. Esempi come quello appena visto veicolano significati non resi esplicitamente nelle espressioni corrispondenti. È però possibile, mediante una catena inferenziale di implicazioni successive, ricavare questo significato. Da questo punto di vista il proferente veicola la sua intenzione (la richiesta di chiudere la

[1]Per la teoria degli atti linguistici (*speech-acts*) si rimanda all'articolo di Austin [1] ed alla letteratura in merito.

porta) basandosi sul fatto che l'ascoltatore sia in grado di aggiungere le appropriate implicazioni aggiuntive, quali:

> –*The speaker has brought a gale to my attention.*
> –*The speaker has some attitude to the gale.*
> –*Gales, winds, etc. lower the apparent temperature by the wind chill factor.*
> –*The gale is making the speaker cold and therefore uncomfortable.*
> –*There's an open window.*
> –*Open windows admit draughts, winds, gales to rooms.*
> –*The open window is probably responsible for the gale the speaker is feeling.*
> –*I am nearer the open window than the speaker and could easily close it.*
> –*The speaker wants me to close the open window.* [3, Pgg. 6-7]:

È stato obiettato a questo tipo di approccio alla pragmatica (definito *radical inferential approach*) che proferimenti quali quello visto in esempio sono di fatto codificati convenzionalmente. Un ragionamento radicale come quello mostrato sopra può essere possibile solamente quando l'ascoltatore ha piena conoscenza e comprensione di ogni implicazione che costituisce la catena inferenziale, ma ciò non è necessario. Nella maggior parte delle occasioni queste forme convenzionali indirette sono memorizzate come fossero degli stratagemmi (*gambit*) conversazionali, da adattarsi direttamente alla situazione a seconda delle necessità. Sarà questo secondo approccio, un "approccio convenzionale alla pragmatica", quello condiviso durante lo sviluppo del sistema di comprensione del linguaggio naturale affrontato nel presente elaborato.

È tuttavia da ricordare come il contesto, nonostante le convenzioni, possa sempre cambiare il significato sottointeso. Può inoltre capitare che si scopra essere necessario disambiguare una forma linguistica che convenzionalmente codifica diverse possibili intenzioni. Simili problematiche saranno affrontate nel seguito della trattazione.

1.3 Il ragionamento abduttivo

> *We assume that any NLP system capable of interpreting the intentions behind utterances will need to be able to understand them relative to an evolving context, and that this will involve inference of a non-deductive, indeed an abductive nature.* [3, Pgg. 6-7]

Come evidenziato dall'estratto riportato qui sopra, l'abduzione[2] sembra essere il procedimento inferenziale comunemente utilizzato nel ragionamento umano. È però un metodo non logicamente valido (quanto non lo è l'induzione) come si può facilmente notare:

Deduzione:	Induzione:	Abduzione:
$P \Rightarrow Q$	x è P	x è Q
x è P	x è Q	$P \Rightarrow Q$
$[x$ è $Q]$	$[P \Rightarrow Q]$	$[x$ è $P]$

L'abduzione, mediante delle assunzioni, tenta di spiegare le osservazioni registrate. Ad esempio, se si osserva una lampadina guasta, si ipotizzano quali condizioni si siano verificate tali per cui questa abbia smesso di funzionare. Un altro esempio di ragionamento comune in cui ritroviamo un procedimento inferenziale abduttivo è il seguente (si noti il parallelo con il ragionamento deduttivo logicamente corretto):

Deduzione:	Abduzione:
Quando piove la strada è bagnata	La strada è bagnata
Piove	Quando piove la strada è bagnata
[La strada è bagnata]	[Piove]

Durante il processo abduttivo viene determinato ciò che potrebbe implicare le osservazioni registrate ("La strada è bagnata"), ovvero ciò che se fosse vero renderebbe l'osservazione vera[3].

Si vedrà ora una formalizzazione del suddetto processo inferenziale (tratta da [15]) al fine di garantirne una comprensione maggiore, sciogliere alcune ambiguità di fondo e fornire un'impalcatura logica iniziale per la successiva trattazione implementativa.

Formalizzazione del procedimento abduttivo Si dia una conoscenza di base (*knowledge base*) KB che sia un insieme di clausole di Horn[4] ed un insieme A di

[2] Il termine "abduzione" è stato coniato da Charles Sanders Peirce (1839-1914) per differenziarlo dagli altri due tipi di ragionamento conosciuti, quello deduttivo e quello induttivo.

[3] Va da sè notare come ogni osservazione sia banalmente implicata dalle contraddizioni, essendo le contraddizioni implicanti qualsiasi cosa. Queste saranno perciò da escludersi.

[4] Una clausola di Horn è, nel calcolo proposizionale, una disgiunzione di letterali in cui al massimo uno dei letterali è positivo, come ad esempio: $\neg A \vee \neg B \vee \neg C \vee D$ (il numero di letterali

proposizioni elementari (atomi) detti "assunti"(*assumables*) (elementi costitutivi di ogni ipotesi).
– Le osservazioni non andranno ad aggiungersi alla KB, ma verranno formalizzate sotto forma di spiegazioni (*explanations*).
– Uno *scenario* di $\langle KB, A\rangle$ è un sottoinsieme H di A tale per cui $KB \cup H$ è soddisfacibile (*satisfiable*).
– $KB \cup H$ è soddisfacibile se esiste un modello[5] in cui ogni elemento di KB ed ogni elemento di H è vero, e questo accade quando non vi sono sottoinsiemi di H che siano in conflitto con KB.
– Una spiegazione di una proposizione g per $\langle KB, A\rangle$ è quindi uno scenario che, assieme a KB, implica g, ovvero:

$$[H \text{ spiegazione di } g] \Leftrightarrow [H \subseteq A \quad \text{t.c} \quad KB \cup H \models g \ \wedge \ KB \cup H \not\models falso]\ ^{6}$$

– Una spiegazione minimale di g per $\langle KB, A\rangle$ è una spiegazione H di g per $\langle KB, A\rangle$ tale che non esiste un sottoinsieme stretto di H che sia anch'esso spiegazione di g per $\langle KB, A\rangle$ [7].

Esempio Si dia $KB = \{$bronchite $\leftarrow$ influenza, bronchite $\leftarrow$ fumare, tossire $\leftarrow$ bronchite, ansimare $\leftarrow$ bronchite, febbre $\leftarrow$ influenza, febbre $\leftarrow$ infezione, malDiGola $\leftarrow$ influenza, Falso $\leftarrow$ (fumare $\wedge$ nonFumatore)$\}$ con $A = \{$fumare, nonFumatore, influenza, infezione$\}$.
Se si osserva una persona ansimare, esistono due spiegazioni minimali:

$$g = \text{ansimare}, \quad H_1 = \{\text{influenza}\},\ H_2 = \{\text{fumare}\}$$

---

può essere qualsiasi, volendo anche nessuno). La condizione di positività di un elemento permette di scrivere le clausole come implicazioni (ad es: $(A \wedge B \wedge C) \Rightarrow D$).

[5] La teoria dei modelli è lo studio delle classi delle strutture matematiche dal punto di vista della logica matematica. Gli oggetti di studio sono quindi i modelli delle teorie in formalizzati nel linguaggio della logica matematica. Un insieme di enunciati formalizzati è uno dei componenti che formano le teorie. Un modello di una teoria è una struttura che soddisfa gli enunciati di quella teoria.

[6] Il simbolo $\models$, detto "doppio *turnstile*" è il simbolo di implicazione semantica, e denota una equivalenza logica. Il simbolo $\vdash$ (*turnstile*) denota invece l'implicazione sintattica, ovvero la deducibilità del conseguente rispetto all'antecedente (il primo può essere provato a partire dal secondo in un qualche specifico sistema formale).

[7] Per quanto visto qui ed ulteriori sviluppi riguardo la formalizzazione del procedimento abduttivo cfr.[15].

In questo caso le due spiegazioni H_1 ed H_2 implicano bronchite e tosse. Vediamo altri esempi di osservazioni:

$g =$ansimare $\wedge$ febbre, $H_1 = \{$influenza$\}$, $H_2 = \{$fumare, infezione$\}$

$g =$ansimare $\wedge$ nonFumatore, $H_1 = \{$influenza, nonFumatore$\}$[8]

1.4 Abduzione e pragmatica nell'interazione uomo-robot

Come già evidenziato, ciò che avviene durante la comprensione dei proferimenti del linguaggio naturale è l'applicazione del ragionamento abduttivo. Ad esempio, le implicature conversazionali vengono comprese quando viene costruita una spiegazione H che permette di interpretare coerentemente e/o cooperativamente tale proferimento [9]. Non essendo però l'abduzione logicamente valida, le implicature conversazioniali sono da considerarsi *defeasible*, ritrattabili (o "rivedibili").

Il presente elaborato parte dagli assunti teorici qui presentati al fine di progettare un sistema in grado di svolgere un ragionamento abduttivo nell'interazione con un utente umano. Specificamente, si tenterà di definire un sistema in grado di identificare, in base a delle informazioni contestuali e mediante l'input dell'utente, il comando o la richiesta che l'utente sottointende con il suo proferimento o, più semplicemente, la reazione che l'utente nel suo interagire con il robot ha intenzione (o avrebbe intenzione) di provocare in esso. Ad esempio, se l'utente interpellando il robot affermasse una frase del tipo "si ghiaccia qui dentro", sarebbe compito del robot capire, in base al contesto (ad esempio, se fosse inverno ed il robot avesse la possibilità di accendere i caloriferi), quale sia la reazione adeguata (in questo caso accendere i caloriferi).

Vista la complessità e la vastità della problematica l'implementazione si limiterà ad uno studio di un caso ristretto, un *toy model* che permetta di esplicitare e testare le dinamiche del sistema senza dover affrontare in una sola volta tutte le problematiche del caso. Nell'ultimo Capitolo verrà poi affrontata la tematica nella sua totalità

[8] Per altri esempi ed ulteoriori approfondimenti cfr.[15].

[9] Un esempio di implicatura conversazionale la cui comprensione è spiegata secondo questo meccanismo è la seguente: A:"Ho finito la benzina"–B:"C'è un garage dietro l'angolo"[3].

cercando di proporre direzioni di ricerca forse in grado di aggirare le limitazioni del modello qui proposto.

Capitolo 2

Definizione del problema

2.1 Presentazione del modello

Si immagini un contesto di conversazione ordinario. Si è in una stanza della propria casa, con un'altra persona, e si è in un determinato stato fisico o mentale quale potrebbe essere il provare molto caldo. Si può quindi comunicare all'altra persona la situazione, in questo caso di fastidio, con una frase del tipo "ho molto caldo". Verosimilmente si avrà una risposta da parte dell'altra persona, probabilmente volta a rimediare alla situazione di fastidio, quale potrebbe essere "vuoi che apra la finestra?", "vuoi che ti porti un bicchiere d'acqua?". Potrebbe inoltre esserci in risposta direttamente il compimento dell'azione corrispondente, nel caso l'ascoltatore la riconosca come appropriata ed adeguata alla situazione. Che l'intenzione del proferente fosse quella di farsi aprire la finestra o di farsi portare un bicchier d'acqua, o che questi volesse semplicemente comunicare la sua situazione, l'ascoltatore ha applicato il ragionamento abduttivo (Sezione 1.3) per identificare un significato ulteriore, sia stato questo una richiesta velata, un comando espresso non direttamente o addirittura un'intenzione non esplicitamente pensata dal proferente ma desiderata (si immagina sia piacevole per lui l'apertura della finestra). Il sistema qui progettato dovrà quindi permettere ad un robot di sostituirsi idealmente all'ascoltatore senza che vi siano differenze in termini di reazione comportamentale al proferimento.

Nel progetto che qui si andrà a discutere il processo di interazione uomo-robot è stata diviso in quattro fasi principali: identificazione, inferenza, interrogazione e reazione (come illustrato alla Figura 2.1), in modo tale che il lavoro complessivo fosse funzionalmente distribuito in quattro macro-processi in qualche modo analoghi

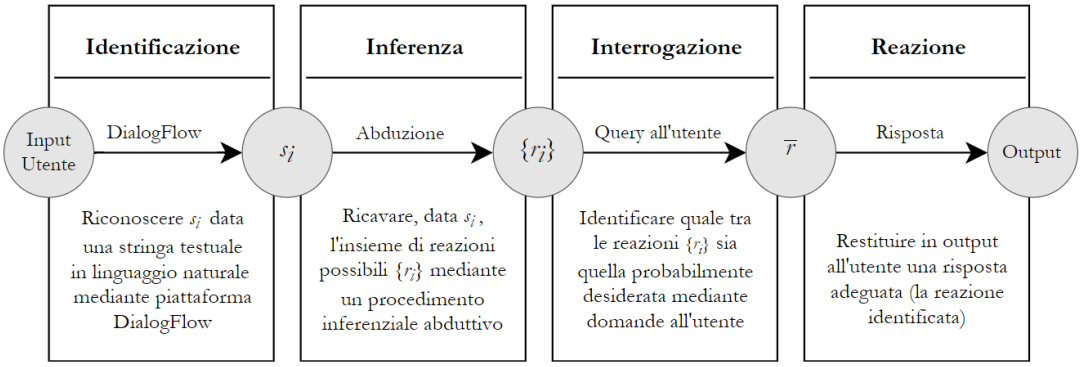


Figura 2.1: Funzionamento generale del sistema di identificazione e reazione

a quelli che esegue l'essere umano in interazioni di questo genere.

Identificazione Durante la fase di identificazione viene svolta un'analisi della stringa in linguaggio naturale proveniente dall'utente. Si immagina infatti che l'apparato "uditivo" *speech-to-text* del sistema robotico restituisca una stringa coerente e per quanto possibile simile a quanto pronunciato dall'utente. Compito di questo primo modulo, il cui procedimento è descritto ed analizzato esaustivamente nel Capitolo 3, è quello di associare l'input dell'utente ad una determinata situazione (quale potrebbe essere l'avere caldo dell'esempio precedente). La fase di identificazione è quindi la fase di riconoscimento del significato letterale del proferimento, o meglio, di "comprensione"del proferimento stesso. È infatti mediante questa fase che una stringa di testo guadagna un significato, e tale significato è guadagnato in quanto rivelatore di una determinata situazione. Si potrebbe quindi dire che, più che comprendere un significato dall'input e da esso dedurne la situazione, il sistema riconosca solo una serie determinata di significati (un "alfabeto semantico" S di situazioni s_i), significati condivisi da molteplici stringhe di testo in linguaggio naturale aventi caratteristiche comuni. Per la fase di identificazioni ci si è serviti della piattaforma DialogFlow (illustrata alla Sezione 3.1) la quale, mediante una serie di *training phrase* anch'esse in linguaggio naturale definite per ogni possibile $s_i \in S$, è in grado di associare le s_i agli input correlati alle rispettive *training phrase*.

Inferenza Durante la fase di inferenza mediante il procedimento inferenziale abduttivo, presentato nel Capitolo precedente, vengono identificate le possibili reazioni

r_i che potrebbero essere eseguite in risposta alla situazione identificata. Per la modellazione del procedimento si è assunto un approccio convenzionale (Sezione 1.2) rispetto alle forme linguistiche codificanti intenzioni o significati diversi da quello esplicito. Pertanto il sistema non dovrà, dalla singola situazione identificata, costruire un sistema di inferenze logiche che colleghino il proferimento al significato aggiuntivo (abbiamo visto come nemmeno l'essere umano attui un processo del genere ma spesso si limiti ad una mappatura convenzionale dei proferimenti) ma semplicemente memorizzare il sistema di convenzioni che conducono a determinati significati, dati determinati contesti o situazioni. Si è scelto quindi di modellare il procedimento abduttivo nel seguente modo:

- Si diano un insieme S di situazioni s_i ed un insieme R di reazioni r_i.
- KB sarà del tipo $KB = \{r_i \leftarrow s_j, ...\}$
- A è da identificarsi con l'insieme R
- Dato un input dell'utente g, verranno inferite n spiegazioni $H_l, l \in [1, n]$

Per come è stato costruito KB ogni spiegazione H_l conterrà un solo elemento di R. Per questo motivo si identifica come risultato dell'ipotesi inferenziale H_l la singola r_i. L'insieme delle n reazioni di R inferite sarà poi soggetto di analisi, al fine di capire quale sia la reazione voluta dall'utente o quella maggiormente desiderata (o desiderabile), vista la situazione, rispetto al contesto di proferimento. Nella Tabella 2.1 si può vedere un esempio di KB in forma di tabella. Il tipo di tabella mostrata rappresenta un determinato contesto, nel quale alcune implicazioni sono codificate da determinate situazioni. La tabella in questione, si potrebbe dire, è la "tabella delle convenzioni linguistiche"che si hanno in un determinato contesto linguistico, una tabella contestuale che permette di decodificare i significati non letterali mediante il procedimento abduttivo.

KB		
$r_1 \leftarrow s_1$	$r_2 \leftarrow s_1$	
	$r_2 \leftarrow s_2$	$r_3 \leftarrow s_2$
	$r_2 \leftarrow s_3$	
$r_1 \leftarrow s_4$		

Tabella 2.1: Esempio di KB in tabella contestuale (con $A \equiv R = \{r_1, r_2, r_3\}$)

Interrogazione Per fornire un criterio di scelta delle reazioni tra quelle inferite dell'insieme $\{r_i\}$ si è deciso di inserire, nella tabella contestuale, una mappatura situazioni-probabilità che, riassumendo quanto visto nella Tabella 2.1, fornisca anche un metodo per disambiguare tra possibili spiegazioni. Sono quindi date una serie di probabilità del tipo $P(s_i \Rightarrow r_j)$ grazie alle quali il sistema è in grado di discernere quale reazione sia più appropriata, anche con l'ausilio di successivi scambi di informazione.

S	r_1	r_2	r_3
s_1	$P(s_1 \Rightarrow r_1)$	$P(s_1 \Rightarrow r_2)$	0
s_2	0	$P(s_2 \Rightarrow r_2)$	$P(s_2 \Rightarrow r_3)$
s_3	0	$P(s_3 \Rightarrow r_2)$	0
s_4	$P(s_4 \Rightarrow r_1)$	0	0

Tabella 2.2: Esempio di tabella contestuale situazioni-probabilità (con 0 sono indicate le rispettive probabilità $P(s_i \Rightarrow r_j)$ nulle)

Nella Figura 2.2 è illustrato lo schema logico del processo di interrogazione: dopo l'identificazione dell'input vi è un processo di valutazione probabilistica delle possibili inferenze che porterà o ad una richiesta di maggiori informazioni –mediante lo stabilirsi di una conversazione con l'utente– o l'esecuzione della reazione inferita (dinamica trattata nel Capitolo 5).

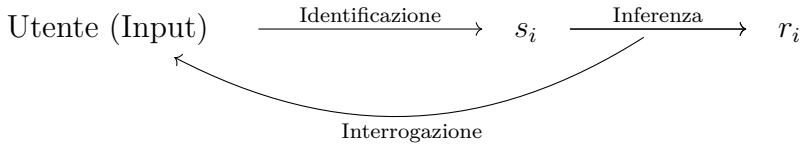


Figura 2.2: Schema logico del processo di interrogazione del sistema

Poniamo infatti che, sempre tenendo presente i dati utilizzati alle Tabelle 2.1 e 2.2, sia stata identificata la situazione s_2. Dopo la fase di inferenza il sistema si ritrova con due possibili reazioni $\{r_2, r_3\}$. Nella fase di interrogazione verrà allora posta all'utente una domanda per capire quale delle due sia la reazione $\bar{r}$ desiderata.

Reazione L'ultima fase, quella di reazione, consiste nell'esecuzione della reazione scelta tra quelle inferite. Nel lavoro qui presentato però, come è già stato detto, non si affrontano tematiche legate alla realizzazione fisica del robot. La fase di reazione

per il sistema qui sviluppato consisterà quindi in una semplice comunicazione in output all'utente del tipo "Penso eseguirò l'azione $\bar{r}$".

2.2 Selezione delle reazioni

Si è qui tentato uno sviluppo basato su tre possibili reazioni $R = \{r_1, r_2, r_3\}$, aventi alcune situazioni s_i in comune. Il sistema sviluppato avrà il compito di capire in quale situazione $s_i \in S$ (con S insieme delle situazioni possibili) si trova durante la sua interazione con l'utente, quali sono le possibili r_i implicate dalla s_i rilevata e con quale probabilità. Questi dovrà quindi scegliere se e quale r_i eseguire, oppure se porre alcune domande all'interlocutore per discernere quale r_i sia quella desiderata o probabilmente desiderata. Nella Tabella 2.2 si è vista una possibile tabella contestuale situazioni-probabilità, nella quale si possono appunto notare due situazioni comuni: s_1 rispetto a r_1 ed r_2 ed s_2 rispetto a r_2 ed r_3. Tenendo presente il contesto definito, si sono cercate una serie di possibili azioni che il sistema robotico potrebbe eseguire in reazione ad alcune frasi comuni pronunciate dall'utente, in modo da poterne poi selezionare tre adatte allo scopo:

- "Ho caldo"
  $R = \{$aprire la finestra, portare un bicchiere d'acqua, accendere il condizionatore$\}$
- La stanza è buia"
  $R = \{$tirare le tende, aprire le persiane, accendere la luce$\}$
- "E' tutto così in disordine"
  $R = \{$fare il bucato, stendere, piegare o stirare dei vestiti, fare il letto, pulire per terra, spolverare, buttare via l'immondizia$\}$
- "Ho voglia di cucinare qualcosa"
  $R = \{$apparecchiare la tavola, prendere qualcosa dal frigo, fare la spesa, accendere il forno, portare il ricettario o cercare la ricetta$\}$
- "Ho fame"
  $R = \{$cucinare qualcosa, prendere qualcosa dal frigo, fare la spesa, ordinare cibo da asporto$\}$
- "Devo andare a lavorare"
  $R = \{$portare lo zaino o la borsa, preparare la borsa o lo zaino$\}$

- "Ho voglia di rilassarmi"
  $R = \{$accendere lo stereo o riprodurre della musica, fare il letto, accendere la televisione$\}$

A partire dall'insieme di reazioni così trovate si è scelto di lavorare con il seguente insieme:

$$R = \{r_1, r_2, r_3\} \tag{2.1}$$

- con $r_1 \triangleq$ "aprire la finestra",
- $r_2 \triangleq$ "portare una bottiglia d'acqua",
- $r_3 \triangleq$ "accendere la televisione".

La scelta è stata motivata dalla praticità del set di reazioni. Intuitivamente infatti è facile che r_1 ed r_2 siano legate da una causa abbastanza forte quale l'avere caldo. È però altrettanto vero che ognuna di esse presenta anche delle cause forti peculiari differenti, quali per una potrebbe essere il dover sporgersi dalla finestra e per l'altra l'avere sete. Si noti inoltre come r_3 appaia invece meno legata alle prime due. Con la scelta di queste tre reazioni è quindi facile che il *toy model* definito permetta di apprezzare ogni dinamica tipica del sistema. Infatti sarà facile che vi sia una rigida esclusione o di r_3 o di r_1 ed r_2 in seguito all'identificazione di una particolare s_i, vista la loro differenza. Avendo invece alcune cause comuni è facile che a volte il sistema si trovi a dover discernere o interrogare l'utente per capire quale reazione tra r_1 ed r_2 adottare.

2.3 Raccolta dei dati

La raccolta di dati ha riguardato tutta la fase preparatoria del progetto. Ciò è stato orientato fondamentalmente a due scopi: trovare il maggior numero di situazioni s_i legate alle reazioni scelte R ed avere un *corpus* di frasi associate ad ogni situazione dell'insieme designato S. Il *corpus* sarà infatti necessario al fine di identificare la situazione s_i a partire dall'input vocale. Per la raccolta dati sono stati utilizzati questionari a risposte aperte realizzati mediante la piattaforma gratuita Google Forms.

2.3.1 Primo Form

Descrizione del Form: "Per il mio progetto di tesi triennale devo raccogliere una serie di cause collegate ad intenzioni ed azioni comuni. Nel presente questionario è richiesto perciò di elencare almeno dieci possibili cause per domanda."

Domande:

- Immagina di essere in casa tua e di voler aprire la finestra. Per quali possibili cause potresti volerlo fare?
- Immagina di essere in casa tua e di voler prendere una bottiglia d'acqua. Per quali possibili cause potresti volerlo fare?
- Immagina di essere in casa tua e di voler accendere la televisione. Per quali possibili cause potresti volerlo fare?

Soggetti: il Form sopra descritto è stato sottoposto a soggetti aventi tra i 19 ed i 55 anni d'età, l'85% attorno ai 20 anni ed il 15% attorno ai 50 anni.

Risultati: nelle Tabelle A.1, A.2 ed A.3 in Appendice sono riportati i risultati delle tre domande. La percentuale indicata è data dalla frequenza di ogni s_i relativamente al totale di situazioni della r_i corrispondente. Si sono quindi selezionate 21 situazioni delle quali 6 comuni tra due delle tre reazioni (Tabella 2.3).
Avendo le r_i un valore assimilabile a quello di "intenzioni" e le s_i a quello di "cause", si è spesso scelto di escludere quelle s_i che avevano un valore di "intenzione" molto forte. Si è infatti cercato, ove possibile, di seguire lo schema minimale di $Causa \rightarrow Intenzione$ (es: "Ho caldo"→ "Apro la finestra") evitando catene del tipo $Causa \rightarrow I_1 \rightarrow I_2 \rightarrow ...$ (es: "Ho i vetri della finestra sporchi"→ "Devo lavare i vetri"→ "Apro la finestra").

2.3.2 Secondo Form

Descrizione del Form: "Per il mio progetto di tesi triennale devo raccogliere un *corpus* di frasi evidenzianti situazioni comuni.
Nel presente questionario è richiesto perciò di elencare cinque possibili frasi che potrebbero essere utilizzate nei contesti indicati per manifestare e/o comunicare la situazione in cui ci si trova.

S			r_1	r_2	r_3
s_1	$\triangleq$	Avere caldo	10%	2%	
s_2	$\triangleq$	Insetto nella stanza	7%	2%	
s_3	$\triangleq$	Piante fuori da bagnare	4%	8%	
s_4	$\triangleq$	Sapere che tempo ci sia fuori	3%		3%
s_5	$\triangleq$	Aria da cambiare	10%		
s_6	$\triangleq$	Rumore/stimolo da fuori	5%		
s_7	$\triangleq$	Odore sgradevole	4%		
s_8	$\triangleq$	Odore di gas	2%		
s_9	$\triangleq$	Sentirsi soffocare	2%		
s_{10}	$\triangleq$	Avere sete		12%	
s_{11}	$\triangleq$	Andare a fare esercizio fisico		5%	
s_{12}	$\triangleq$	Mancanza di acqua a tavola		4%	
s_{13}	$\triangleq$	Fuoco da spegnere in casa		4%	
s_{14}	$\triangleq$	Mancanza di acqua corrente		4%	
s_{15}	$\triangleq$	Dover cucinare	3%		1%
s_{16}	$\triangleq$	Volersi informare sull'attualità			14%
s_{17}	$\triangleq$	Voler guardare un film/serie tv			10%
s_{18}	$\triangleq$	Aver bisogno di svagarsi/rilassarsi			9%
s_{19}	$\triangleq$	Essere annoiati			7%
s_{20}	$\triangleq$	Non riuscire a dormire			3%
s_{21}	$\triangleq$	Troppo silenzio			2%

Tabella 2.3: Analisi risultante dal Primo Form

Ad esempio: per la situazione 'guasto alla macchina' possono essere usate 'la macchina si è rotta', 'il motore mi ha abbandonato'...
N.B.: se si è inserito 'la macchina si è rotta' evitare le frasi pressoché identiche quali 'il veicolo si è rotto', 'la macchina si è guastata' ecc..."

Osservazione: Sono stati creati tre Form aventi ognuno 21 domande, differenti tra di loro per una componente VAR:
- nel primo: $VAR =$ "Con te c'è un tuo familiare",
- nel secondo: $VAR =$ "Con te ci sono degli ospiti",
- e nel terzo: $VAR =$ "Con te c'è il tuo maggiordomo".

Questa variazione è stata introdotta per tentare di avere nelle risposte diversi registri linguistici a seconda delle persone immaginate come presenti nella stanza.

Domande:

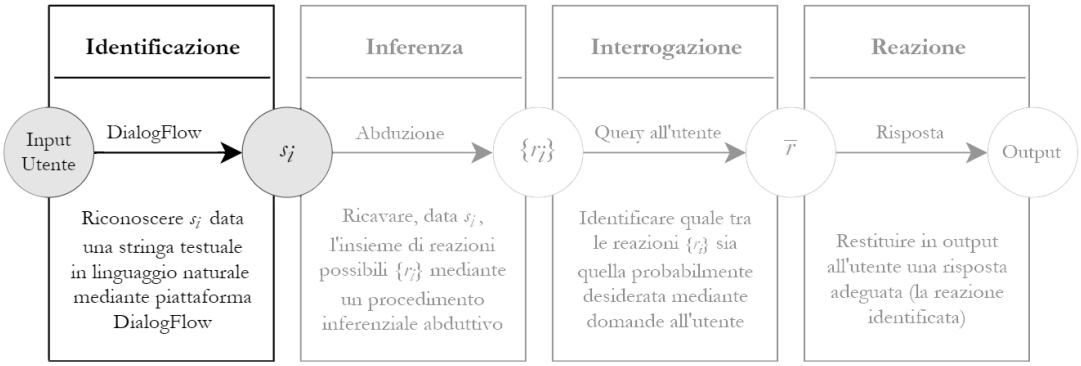
1. Immagina di essere in casa e di avere caldo. VAR
2. Immagina di essere in casa e di vedere un insetto nella stanza. VAR
3. Immagina di essere in casa e di notare che le piante fuori sono da bagnare. VAR
4. Immagina di essere in casa e di non sapere quale sia il tempo fuori di casa. VAR
5. Immagina di essere in casa e che nella stanza in cui ti trovi l'aria sia da cambiare. VAR
6. Immagina di essere in casa e che da fuori provenga un rumore o uno stimolo insolito . VAR
7. Immagina di essere in casa e che ci sia un odore sgradevole. VAR
8. Immagina di essere in casa e che ci sia odore di gas. VAR
9. Immagina di essere in casa e che ti manchi l'aria. VAR
10. Immagina di essere in casa e di avere sete. VAR
11. Immagina di essere in casa e di dover uscire per andare a fare esercizio fisico. VAR
12. Immagina di essere in casa e che manchi l'acqua a tavola. VAR
13. Immagina di essere in casa e che si sia sviluppato un piccolo fuoco libero. VAR
14. Immagina di essere in casa e che ci sia stato un guasto alle tubature tale per cui non c'è più acqua corrente. VAR
15. Immagina di essere in casa e di doverti metterti a cucinare. VAR
16. Immagina di essere in casa e di doverti informare sugli eventi di attualità. VAR
17. Immagina di essere in casa e di voler guardare un film o una serie TV. VAR
18. Immagina di essere in casa e di aver bisogno di rilassarti o di svagarti. VAR
19. Immagina di essere in casa e di essere annoiato. VAR
20. Immagina di essere in casa e di non riuscire a dormire. VAR
21. Immagina di essere in casa e che ci sia troppo silenzio. VAR

Soggetti: il Form sopra descritto è stato sottoposto a soggetti aventi tra i 20 ed i 50 anni d'età, il 6% di 20 anni, il 41% di 21 anni, il 47% di 22 anni e il 6% di 50 anni.

Risultati: Il *corpus* ricavato consta di 85 frasi per ognuna delle 21 situazioni, per un totale di 1785 frasi. Per completezza è stato riportato in Appendice B.

Capitolo 3

Identificazione



3.1 La piattaforma DialogFlow

DialogFlow è una piattaforma online basata sullo sviluppo di interfacce conversazionali a base testuale (come i *chatbot* [5]) che permette di modellare risposte ad input testuali, identificandone il contenuto con tecniche di *machine learning*. Fornendo un *training set* di frasi che l'utente potrebbe pronunciare durante la sua interazione con l'interfaccia, questo strumento permette di analizzare e comprendere ciò che viene definito *intent* della frase in input, al fine di restituire in output la risposta più utile a seconda dei casi.

Specificamente, Dialogflow permette di creare un *agent*, mediante il quale implementare un modello di comprensione del linguaggio naturale. Mediante costrutti chiamati *entity* è possibile definire la modalità di estrapolazione dei dati presenti negli enunciati in input, lavorando per tipologia di dato anziché sulle espressioni

specifiche. Mediante i *context* si può invece tener traccia dello stato della conversazione nel procedere del dialogo. Come mostrato nella figura 3.1 è inoltre possibile integrare il modello con le librerie *Actions on Google* o un *webhook* personalizzato mediante le impostazioni di *fullfillment* [13].

3.1.1 Funzionamento della piattaforma

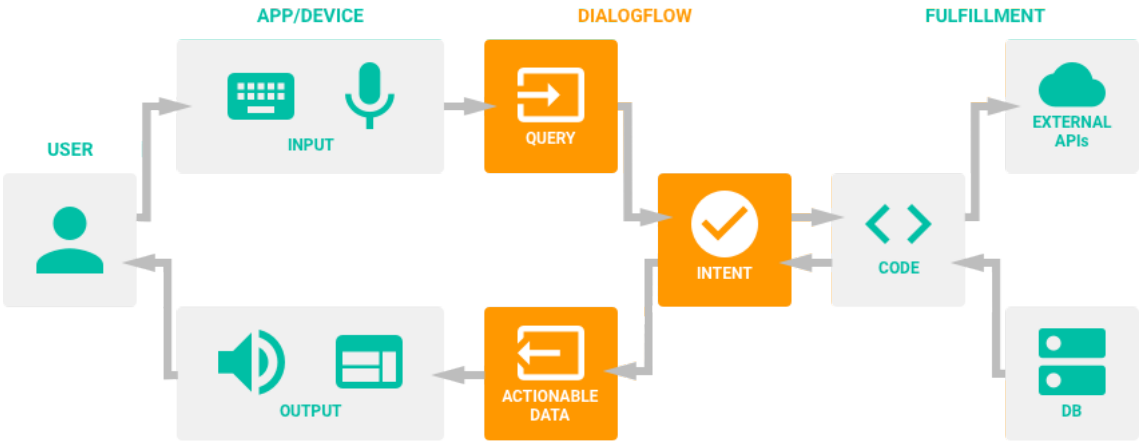


Figura 3.1: Schema generale di funzionamento di un *agent* in DialogFlow (immagine da dialogflow.com/docs/agents).

***Intent* e *context*:** *Intent* e *context* sono i due strumenti chiave che la piattaforma fornisce per modellare il comportamento dell'interfaccia. Gli *intent* collegano ciò che un utente ha detto all'azione che dovrebbe essere successivamente intrapresa dall'interfaccia. I *context* invece sono stringhe che servono a tener conto di ciò che l'utente ha detto precedentemente, al fine di discernere tra i possibili diversi significati che una frase pronunciata potrebbe avere in funzione delle richieste precedenti. Quando DialogFlow riceve in input una stringa testuale dovuta ad una richiesta dell'utente, questa viene classificata per determinare a quale degli *intent* conosciuti appartenga[1]. Si può decidere infatti a quali condizioni un determinato *intent* possa essere identificato fornendo la lista di *context* attivi necessari affinché ciò possa avvenire;

[1]Viene fornito anche il *default fallback intent* per gestire le risposte ad input non corrispondenti ad alcun *intent* precedentemente definito.

allo stesso modo un *intent* può a sua volta attivare uno o più *context*. Grazie a questo meccanismo si può arrivare a creare macchine a stati finiti che gestiscono estesi e complessi scambi testuali.

***Entity*:** Oltre alla logica *intent-context* DialogFlow permette di utilizzare le *entity*, grazie alle quali si possono estrarre i valori di determinati parametri associati dagli input testuali (enunciati direttamente nella lingua naturale), avendo quindi una corrispondenza biunivoca tra entità e tipo di dato corrispondente. Il meccanismo è estremamente flessibile poiché al fine del suo funzionamento non è necessaria una struttura di *entity* completa ed esaustiva. DialogFlow permette infatti di estrarre i dati corrispondenti senza bisogno di classificarne ogni possibile tipo, rendendo possibile la definizione delle *entity* personalizzate, e l'integrazione di quelle predefinite, solamente rispetto ai dati che interessano il contesto di applicazione.
Vengono forniti diversi gruppi di *entity* di sistema, classificati in tre macro-famiglie: le *system mapping entities*, le *system enum type entities* e le *system composite entities*. Ad esempio l'*entity* `@sys.date` – la quale estrae da stringhe in linguaggio naturale quali "tra tre giorni" o "il primo Marzo del 2018" il valore di riferimento `2018-03-01`[2] – appartiene al primo gruppo, mentre l'*entity* `@sys.color` – che estrae da stringhe quali "scarlatto", "cremisi" o "carminio" il termine "rosso"– appartiene al secondo.

***Agent* e *machine learning*:** L'*agent* è il modulo complessivo di *Natural Language Understanding* (NLU) –contenente gli *intent*, le *entity* e tutto il resto– che trasforma le richieste in linguaggio naturale dell'utente in dati utilizzabili (qualora l'input corrisponda ad uno degli *intent* definiti all'interno dell'*agent*).
Ogni *agent* ha due parametri principali per gestire i meccanismi di *machine learning* alla base dell'identificazione degli *intent*:

1. *Match Mode*: il parametro specifica quali algoritmi debbano essere utilizzati negli *intent* in cui è abilitato il *machine learning*. Per associare all'input dato l'*intent* corrispondente, sono disponibili due diversi algoritmi di *matching*:

    - *Rule-based grammar matching*: algoritmo accurato sia per piccoli che per grandi *corpus* di *training phrases*, permette inoltre un aggiornamento

[2]Il valore è dato in formato standard ISO-8601

rapido dei *model*. Di contro però rende il *matching* su *corpus* di grandi dimensioni lento[3].

- *ML matching*: un algoritmo accurato per grandi *corpus*. Nonostante consenta un *matching* rapido, diventa impreciso per *corpus* di piccole dimensioni. Inoltre l'aggiornamento dei *model* avviene lentamente[4].

Il *Match Mode* permette di scegliere tra due modalità di integrazione degli algoritmi:

- *Hybrid*: viene tentato prima il *rule-based grammar matching*. Se non è stata trovata una corrispondenza si passa quindi al *ML matching*.
- *ML Only*: viene eseguito solamente il *ML matching*.

2. *ML Classification Threshold*: il parametro specifica la soglia di filtraggio dei risultati del *matching* al fine di eliminare i falsi positivi. Le corrispondenza tra input ed *intent* trovate dagli algorimi hanno un *confidence value* (un coefficiente di similarità) che varia in un intervallo da `0.0` a `1.0`. Se questo valore è inferiore alla soglia definita dal parametro viene restituito, se definito, il *default fallback intent*.

***Actions* e *parameters*:** Le *action* definiscono ciò che avverrà in seguito all'identificazione dell'*intent*. Una *action* può contenere dei *parameter*, il cui valore viene ricavato dall'enunciato in input. Entrambi sono restituiti dall'*agent* in formato JSON {`action: action_name`} e {`parameter_name: parameter_value`}.
Ad esempio, si immagini di definire una *action* chiamata `weather.activity` avente tre *parameter* `activity`, `date-time` ed `address`[5]. Nella Tabella 3.1 si può vedere come viene classificata la Training Phrase "posso nuotare oggi a Londra?".
Nell'Appendice D.2 è possibile leggere la risposta completa, in formato `json`, ad

[3]Può inoltre produrre risultati errati se l'*agent* utilizza l'*entity* `@sys.any` oppure se usa frequentemente l'opzione *Allow automated extension*.

[4]Oltre a ciò l'algoritmo si rivela meno accurato rispetto al precedente se si ha a che fare con estesi o complessi *training phrase template* e/o *composite entity*.

[5] L'esempio sfrutta l'*agent* preimpostato `weather`. L'*action* descritta si trova all'interno dell'omonimo *intent* `weather.activity` (cfr. www.dialogflow.com/docs/prebuilt-agents).

un'interrogazione del tipo "Secondo te posso correre domani a Londra?", contenente i dettagli riguardanti le *action* ed i *parameter*.

parameter_name	*Entity*	parameter_value
activity	@activity	swimming
date-time	@sys.date-time	2018-07-09
address	@sys.location	{"city":"Londra"}

Tabella 3.1: Classificazione della *action* `weather.activity` per la *training phrase* "posso nuotare oggi a Londra?"

***Automated expansion*:** L'*automated expansion* è una tecnica di apprendimento –attivabile per le *entity* definite dallo sviluppatore– che consente a un *agent* di espandere i possibili valori di un *parameter* con altri simili, non esplicitamente elencati nell'*entity*. Se l'input dell'utente contiene un elemento che non è elencato nell'*entity*, analizzando la similitudine tra l'input e gli esempi forniti l'*agent* riconosce allora come *parameter* l'elemento estraneo.
Si consideri, ad esempio, una lista della spesa contenente diversi oggetti quali {`pane`, `burro`, `latte`, `frutta`, `gelato`}. Se in input si ha la frase "Ho bisogno di comprare alcune verdure", {`verdure`} verrà interpretato come valore anche se non incluso nell'*entity* `@item`.

Risposte ed eventi: DialogFlow permette infine di definire un insieme di risposte per ogni *intent*, siano esse predefinite o specificate dallo sviluppatore, ed un sistema di invocazione degli *intent* basato su *event*, i quali consentono appunto di invocare *intent* senza che vi sia in input la corrispettiva richiesta da parte dell'utente.

3.1.2 Integrazione nel progetto

DialogFlow viene utilizzato nel progetto per gestire la fase di identificazione della situazione s_i a partire dalla stringa di testo in input. Le fasi di integrazione della piattaforma all'interno del sistema –che verranno trattate nel seguito dell'elaborato– sono le seguenti:

1. creazione del *corpus* in formato `.csv` e conversione in file `.json`;

2. creazione automatica degli *intent* a partire dai file `.json` e caricamento delle *training phrase*;
3. sviluppo di una interfaccia utente da terminale Windows per testare manualmente l'*agent*;
4. analisi di efficacia mediante *k-fold cross-validation*.

Per l'implementazione dei passi sopracitati sono stati utilizzati *script* in Python integranti le API di DialogFlow[6] (vedi Appendice C).

3.2 Preparazione del *corpus*

Si è creato prima di tutto un *agent* su DialogFlow (chiamato "Situazioni"), privato del *default fallback intent* e di qualsiasi altro *intent*. Si andranno infatti ad inserire gli *intent* personalizzati con le *training phrase* presenti nel *corpus* sfruttando le librerie Python `apiai`. Lo *script* in Python dovrà quindi, presi in ingresso i file `json` con le *training phrase* per ogni *intent*, caricare i dati in esso contenuti nell'*agent* designato. Prima di tutto però bisogna costruire i file `json` a partire dal *corpus* di frasi raccolto.

Un file `settings.json`, come illustrato nel Listing 3.1, contiene impostazioni generali legate all'*agent* creato quali la lingua utilizzata, i *token* legati al *client* ed al *developer* (necessari per collegarsi all'*agent* tramite gli *script* di *training* che si dovranno successivamente utilizzare) ed altri dati di minor interesse, come una lista di *follow-up* in diverse lingue[7].

Mentre i risultati del primo Form sono stati utilizzati per la definizione di S (come visto nella Tabella 2.3), il *corpus* di frasi ricavato dai risultati del secondo Form è stato catalogato in un file `.csv` strutturato come riportato nel Listing 3.2. Come si può infatti notare, è stato definito per ogni situazione di S un *intent* di DialogFlow

[6] Più precisamente si sono utilizzate le `V1 Client Libraries` per Python (il *package* `apiai`), meno recenti (documentazione disponibile alla pagina dialogflow.com/docs/reference). "Api.ai" era infatti il nome iniziale della piattaforma. Le librerie in questione non saranno più disponibili dal 10 Aprile 2019, ogni progetto successivo alla data dovrà quindi utilizzare le `V2 Client Libraries` (per Python il *package* `dialogflow`). Maggiori informazioni sull'SDK di DialogFlow in Appendice C ed alla pagina dialogflow.com/docs/sdks. Per informazioni sul *porting* da `V1` a `V2` consultare la pagina dialogflow.com/docs/reference/v1-v2-migration-guide.

[7] Il file `settings.json` è riportato integralmente in Appendice, alla Sezione D.1.

```
{
   "starting-language": "italian",
   "dialogflow": {
      "agent": "Situazioni",
      "default-language": "it",
      "url": "https://api.dialogflow.com/v1/intents",
      "protocol-version": "?v=20150910",
      "intents-folder": "Dialogflow-Intents",
      "tokens": {
         "Situazioni": {
            "client": "59a5bcd8005d4952a27c115511760663",
            "developer": "c17d9ca5ec2e4dfebd3e8404ef0680ae"
         }
      },
      "followups": {
         "it": {
            "yes": [
               "esatto",
                  (...)
```

Listing 3.1: Estratto dal file `settings.json` utilizzato.

dedicato, per il quale sono state inserite tutte le *training phrase* (catalogandole come `user-input`). Unica eccezione il *default fallback intent* che non ne contiene ed a cui è associata la risposta generica "Non ho capito".

```
      INTENT NAME;PARENT'S INTENT;IS PARENT?;;it
      Default Fallback Intent;;;answer;Non ho capito.
      ;;;;
      avere-caldo;;;user-input;Che caldo!
      avere-caldo;;;user-input;Si muore di caldo
      avere-caldo;;;user-input;Che caldo!}
      (...)
```

Listing 3.2: Estratto dal *dataset* `.csv` utilizzato.

Un primo *script* in Python (`csvToJson.py`, vedi Figura 3.3, algoritmo in pseudocofice nel Listing 3.3) attua la conversione da `.csv` a `.json` del corpus ricavato. Lo script crea un dizionario in cui immagazzina, per ogni *intent* i dati e le *training phrase* necessarie, per poi successivamente scrivere il tutto su file `.json`. La

```
input: file 'dataset.csv'
begin
for intent in 'dataset.csv'
      create intent dictionary
      for row in intent
            store row in intent dictionary
      end for
end for
foreach intent dictionary
      create json File in 'intentFolder'
      store intent dictionary in json file
end foreach
```

Listing 3.3: Pseudocodice di `csvToJson.py`.

creazione del dizionario mappa ogni *training phrase* al rispettivo *intent*. Per ogni *intent* presente nel dizionario verrà creato un file `.json` diverso, per un totale di 22 file ($s_1, ..., s_{21} \in S$ ed il *default fallback intent*) da caricare online durante la fase di addestramento.

Nel Listing 3.4 possiamo vedere come lo *script* restituisca in output il file `.json` relativo all'*intent* (in questo caso `aria-cambiare_it.json`). Non avendo utilizzato alcun *context*, *reponse*, *parameter* o *message* per la definizione dell'intento, le uniche informazioni legate all'*intent* `aria-cambiare` sono presenti nel campo `"userSays"`. in cui vengono riportate tutte le 85 *training phrase*.

3.3 Addestramento dell'*agent*

Un secondo *script* in Python (`intentUpload.py`, vedi Figura 3.3, algoritmo in pseudocodice nel Listing 3.5) carica i 22 file `.json` ($s_1, ..., s_{21} \in S$ ed il *default fallback intent*) direttamente nell'*agent* DialogFlow precedentemente creato. Come si può notare alla Figura 3.2, gli *intent* sono stati creati nell'*agent* specificato ed accessibili mediante la console online fornita da DialogFlow.

Al caricamento del *corpus* è succeduta una fase di allenamento dei singoli *intent* mediante gli algoritmi di *machine learning* predisposti. Nella fattispecie, per l'*agent* "Situazioni" si è scelto l'*hybrid match mode* (vedi Sezione 3.1.1) con un *ML classification threshold* di `0.3`.

```
{
   "name": "aria-cambiare",
   "auto": true,
   "contexts": [],
   "templates": [],
   "responses": [
      {
         "parameters": [],
         "messages": [
            {
               "type": 0,
               "speech": []
            }
         ],
         "defaultResponsePlatforms": {},
         "speech": []
      }
   ],
   "userSays": [
      {
         "myPersonalId": "aria-cambiare-084-it",
         "data": [
            {
               "text": "Non␣si␣pu\u00c3\u00b2␣respirare."
            }
         ],
```

Listing 3.4: Risultato della conversione per l'intent "Aria da cambiare", estratto dal file `aria-cambiare_it.json`.

Terminata la fase di addestramento, l'*agent* è operativo, ed è possibile eseguire i primi test mediante l'interfaccia integrata nella console di DialogFlow. Si è scelto poi di implementare un *client* da terminale Windows, scritto in linguaggio Python, per avere una gestione migliore delle risposte e poter definire diversi algoritmi di identificazione, oltre che per fornire uno *script* implementabile, con le modifiche del caso, su diversi sistemi, rimanendo quindi indipendente dall'esecuzione della console sulla pagina web.

```
input: file 'settings.json', folder 'intentFolder'
begin
AgentName, AgentTokens ← 'settings.json'
create connection to agent(AgentName, AgentTokens)
for json file in 'intentFolder'
   read intent data from json file
   upload intent data to AgentName
end for
```

Listing 3.5: Pseudocodice di `intentUpload.py`.

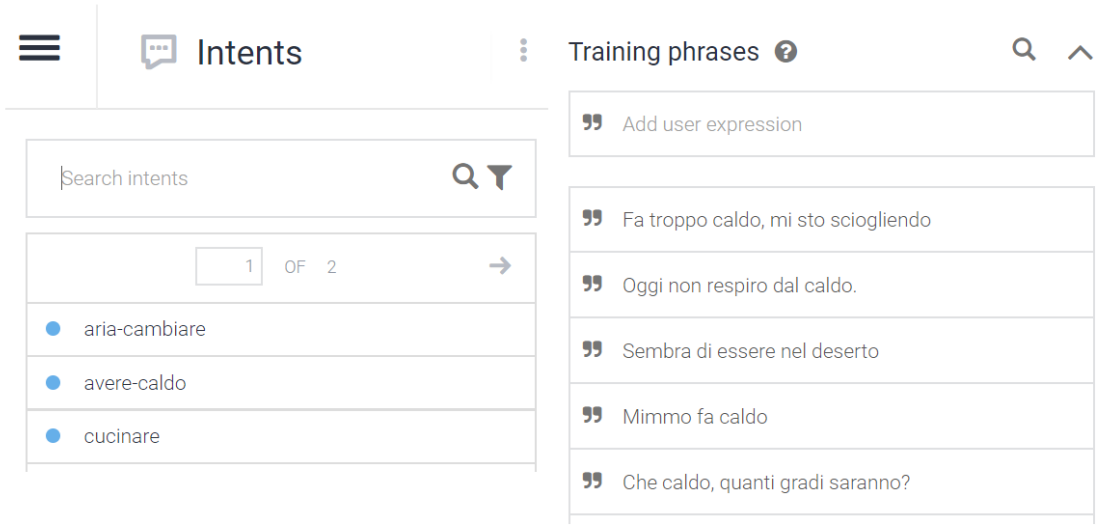


Figura 3.2: *Intent* caricati nell'*agent* DialogFlow e *training phrase* dell'*intent* `avere-caldo` nella console online della piattaforma.

3.4 Sviluppo dell'interfaccia utente

Ai fini di testare le funzionalità dell'*agent* appena addestrato si è sviluppato uno *script* Python (`client.py`, vedi Figura 3.3) per far interagire l'utente da riga comando di Windows direttamente con l'*agent* DialogFlow. Lo *script* sfrutta le librerie `apiai` per collegarsi all'*agent* stabilendo una connessione `ai = apiai.ApiAI(CLIENT_ACCESS_TOKEN)`, passando poi le informazioni della sessione (e l'input dell'utente) alla funzione `getResponseFromDialogflow(ai, user_input)` (algoritmo in pseudocodice nel Listing 3.6) la quale, presa in input una stringa testuale, interroga l'*agent* e ne restituisce la risposta in formato `json`.

```
input: session 'ai', string 'userInput'
begin
response ← query to ai with userInput
create file 'answer.json'
store response in 'answer.json'
```

Listing 3.6: Pseudocodice della funzione `getResponseFromDialogflow`

Il client legge quindi il file così restituito e da esso estrae l'*intent* rilevato (i file di risposta `.json` sono riportati in Appendice D.2). In questo caso l'algoritmo di identificazione è il più semplice possibile: l'*intent*, e quindi la situazione identificata, è quello restituito dall'*agent* in risposta all'input vocale. Alla Sezione 7.2 si è tentato di definire metodi di analisi approfondita che sfruttino in maniera migliore le risposte dell'*agent*.

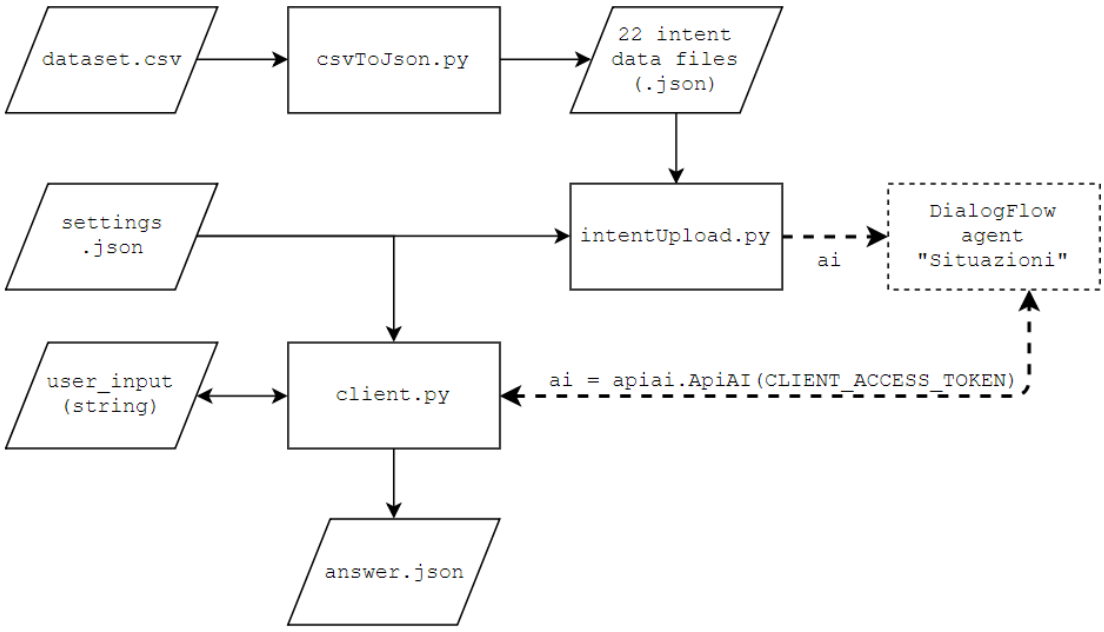


Figura 3.3: Schema generale del processo di addestramento dell'*agent* ed identificazione della situazione (*intent* di DialogFlow). Nella figura i dati sono rappresentati mediante parallelogrammi, i rettangoli rappresentano le operazioni eseguite sui dati mediante *script* in linguaggio Python. L'*agent* DialogFlow è indicato mediante rettangoli tratteggiati, le connessioni ad esso, mediante SDK, sono rappresentate con le linee tratteggiate (grafico realizzato con draw.io).

Capitolo 4

Analisi di efficacia

Come ogni modello *machine learning*, il processo di identificazione basato sulla piattaforma DialogFlow descritta nel Capitolo precedente necessita di essere validato. Una volta sviluppato un *model*, infatti, questi va analizzato misurandone l'accuratezza delle previsioni per input estranei al *corpus* di addestramento. Essendo il procedimento di identificazione della s_i fondamentale per il corretto funzionamento del sistema qui sviluppato, quindi, è di particolare importanza studiare le possibili tipologie di errori ai quali è soggetto. Nel seguito del Capitolo si affronterà un algoritmo preciso di validazione, la *k-fold cross validation*, i cui risultati verranno poi studiati e commentati al fine di valutare il *model* qui utilizzato.

4.1 *k*-fold cross validation

La *k-fold cross validation* è un metodo di validazione statistico utilizzato in ambito *machine learning* al fine di stimare validità e correttezza dei *model* sviluppati per un determinato problema di predizione. È un metodo ottimo perché intuitivo, di facile implementazione ed i cui risultati sono generalmente meno pregiudicati dalle condizioni poste rispetto ad altri metodi [2]. La *cross-validation* stima la correttezza dei risultati basandosi su un *set* di dati in ingresso sul quale il sistema non sia già stato addestrato in fase di *training*. È perciò utilizzato un campione limitato per stimare come il modello si comporti quando viene usato per fare predizioni su dati non visti precedentemente.

La procedura è chiamata *k–fold cross validation* perché il parametro k indica il numero di campioni in cui il *dataset* viene diviso. L'algoritmo si sviluppa nelle

seguenti fasi [16]:

1. Randomizzazione del *dataset*.
2. Suddivisione del *dataset* in k campioni (*folds* appunto).
3. Per ogni campione i-esimo dei k creati:
   (a) Considerare il campione i-esimo come *test data set*,
   (b) Considerare i restanti campioni $(k-1)$ come *training data set*,
   (c) Allenare il *model* con il *training set* e valutare le sue risposte in reazione al *test set*,
   (d) Conservare la valutazione E_i del campione i-esimo ed eliminare il *model* precedentemente creato.
4. Valutare la correttezza E del modello in base alle valutazione E_k sui singoli campioni:

$$E = \frac{1}{k}\sum_{i=1}^{k} E_i$$

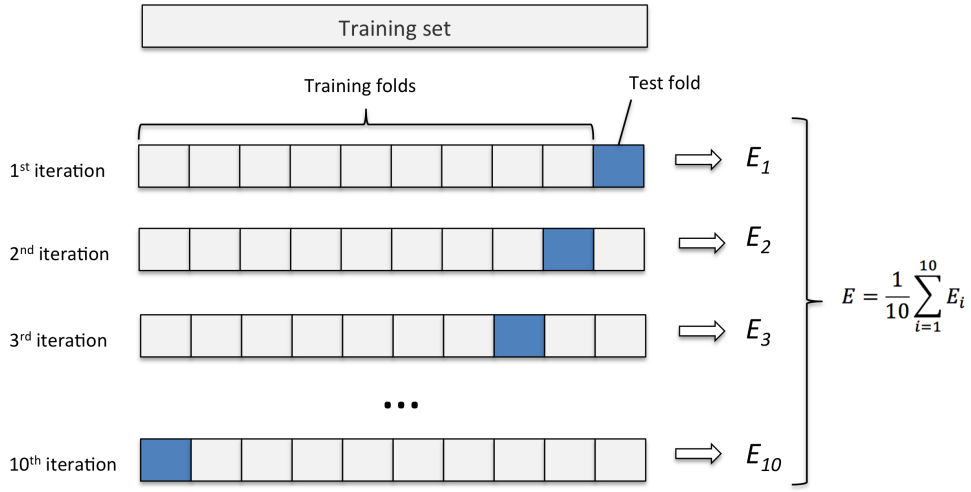


Figura 4.1: 10-*cross fold validation* (immagine da [16]).

Una schematizzazione del processo di suddivisione, analisi e validazione si può trovare nella Figura 4.1, in cui si considera una validazione con $k = 10$ (10-*fold cross validation*). Per il sistema sviluppato si è scelto $k = 5$ come valore del parametro, vista la difficoltà di gestire numeri molto alti mediante la piattaforma DialogFlow (ogni *model* necessita della creazione di un agent completamente nuovo, ogni volta da

collegare manualmente mediante i `client` e `developer token` ad ogni singolo file `settings.json`).

La scelta $k = 5$ non è stata motivata solo da esigenze pratiche, ma ha una utilità di per sé. Non essendoci alcuna regola teorica esplicita, infatti, la scelta per il parametro è di solito $k = 5$ o $k = 10$. A valori di k crescenti, cresce la differenza di dimensioni tra *training* e *test set*, ottenendo quindi risultati bilanciati [12, p.70]. Vi è però da tener conto un altro fattore, ovvero quello della varianza nei risultati E_i del test, che statisticamente aumenta all'aumentare di k. Viene infatti consigliato come default (in assenza di particolari richieste) per sciogliere il *tradeoff* bilanciamento-varianza, una scelta del tipo $k = 5$ o $k = 10$, essendo stato empiricamente evidenziato come questi valori siano adatti a stimare la correttezza di un determinato *model* senza soffrire né di sbilanciamenti eccessivi né di alte varianze nei risultati [8, p.184]. È inoltre preferibile scegliere dei campioni con lo stesso numero di esempi, per fare in modo di avere delle stime E_i più coerenti[2].

4.2 5-fold cross validation sull'*agent*

La preparazione dei file necessari alla validazione si è svolta in diverse fasi, portate a termine mediante *script* Python. Il processo si è articolato nel seguente modo (vedi Figura 4.2):

- Randomizzazione del *dataset* in formato `.csv`.
- Suddivisione del *dataset* randomizzato in cinque *fold*.
- Creazione dei cinque *training set* e dei cinque *test set* per ciascuna delle iterazioni dell'algoritmo di validazione
- Creazione di cinque *agent* dedicati, inizializzazione dei corrispettivi file `settings.json` ed addestramento degli *agent*.
- Implementazione dell'algoritmo di *testing* e calcolo dei risultati.
- Analisi dei dati mediante una stima effettuata sui valori di similarità normalizzata (indicata alla Figura 4.2 come E_{TOT} *table* e riportata in Tabella 4.1) e mediante *confusion matrix* (riportata alla Tabella 4.3).

Randomizzazione per la randomizzazione del *dataset* si è implementato uno *script* Python (`shuffleDataset.py`, vedi Figura 4.2), il quale prendendo in ingresso il file `.csv` originale memorizza ogni riga relativa ad un *intent* in una `list`

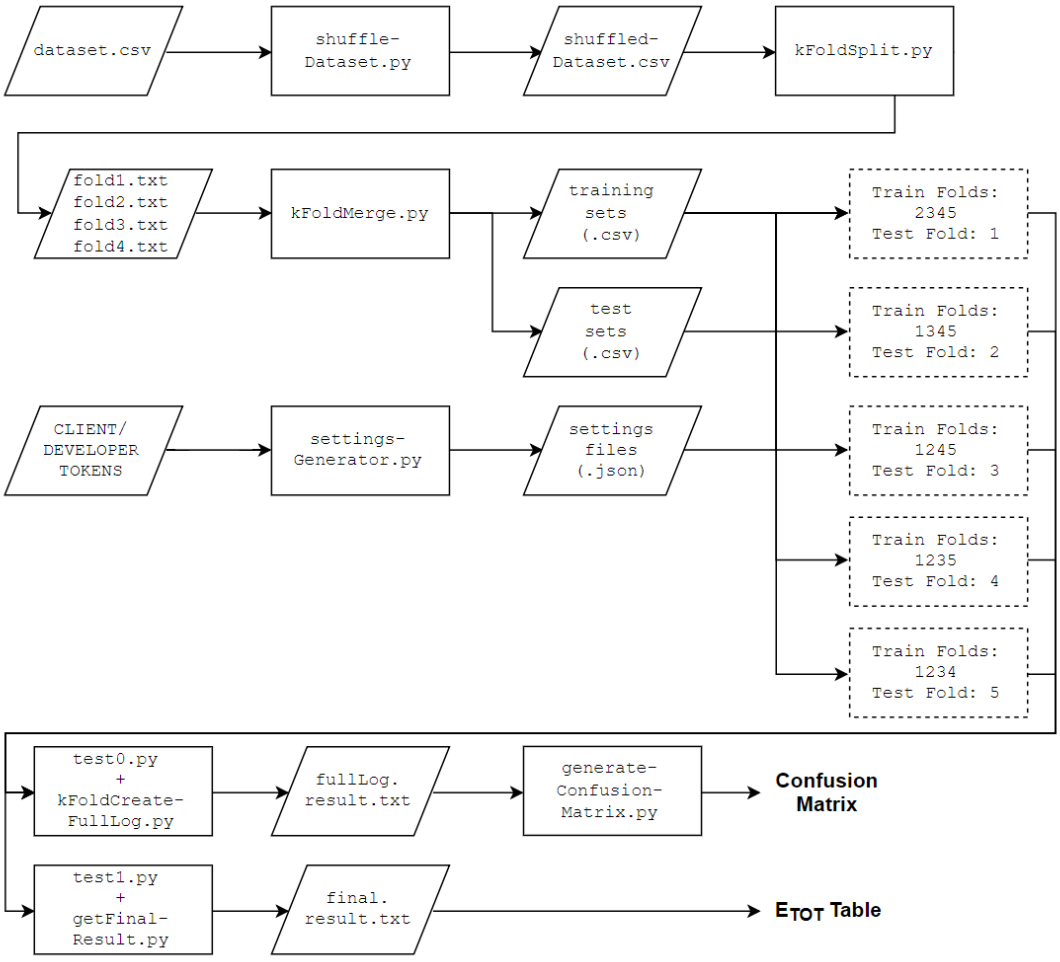


Figura 4.2: Schema generale del processo di validazione. In ogni *agent* DialogFlow è indicato quali *fold* hanno preso parte al *training* e quali al *testing* (vedi i paragrafi dedicati della Sezione 4.2).

di Python differente, per poi randomizzare ognuna di esse e riscriverne il contenuto in formato `.csv`.

Suddivisione per la successiva suddivisione in cinque *fold* si è utilizzato sempre uno *script* Python (`kFoldSplit.py`, vedi Figura 4.2). A causa di 13 risposte vuote all'interno delle 1785, il *corpus* effettivo consta di 1772 frasi. Ogni *intent* ha quindi un numero di *training phrase* legate ad esso che va da un minimo di 80 ad un massimo di 85 frasi. Questa asimmetria tra *training phrase* legate agli *intent* ha

causato alcuni problemi nella divisione in parti uguali dei *fold*. Si è scelto quindi di considerare, per la validazione, solo le prime 80 *training phrase* per ogni *intent* (in riferimento al *dataset* già randomizzato), così da avere una stima bilanciata (come detto precedentemente alla Sezione 4.1).

Preparazione dei *set* e degli *agent* mediante un terzo *script* Python (`kFold-Merge.py`, vedi Figura 4.2) si sono combinati i *fold* in tutte le 5 possibili modalità previste dalla 5-*cross fold validation*. Si sono quindi creati i cinque *agent*, chiamati rispettivamente:

- *KFold*(*Test*1- *Training*2345),
- *KFold*(*Test*2- *Training*1345),
- *KFold*(*Test*3- *Training*1245),
- *KFold*(*Test*4- *Training*1235),
- *KFold*(*Test*5- *Training*1234),

in modo da esplicitare le combinazioni di *fold* utilizzati come *test set* e *training set* (si rimanda a tal proposito sempre alla Figura 4.2). Si sono quindi creati cinque file `settings.json` utilizzati a partire da un *template* (riportato in Appendice D.1) mediante un quarto *script* Python (`settingsGenerator.py`, vedi Figura 4.2) e si è ripetuto il procedimento di addestramento dell'*agent* (visto alla Sezione 3.3) per ognuna delle cinque iterazioni della validazione.

Testing Ogni *fold* utilizzato come *test set* è stato estratto (sono state mantenute solo le singole frasi e non la formattazione `.csv`) in 21 parti sotto forma di file testuali, uno per ogni *intent*. Per eseguire il *testing* vero e proprio si sono poi utilizzati due differenti *script* (`test0.py` e `test1.py`, vedi Figura 4.2).

Lo *script* `test0.py` elabora ognuno dei 21 file testuali e per ognuno di essi scrive un file corrispondente di output.Viene quindi generato, a partire dai 21 file per ognuno dei cinque agent, un unico file `fullLog.result.txt`, mediante lo *script* `kFoldCreateFullLog.py` (vedi Figura 4.2), il quale raccoglie tutti i risultati dello script per i cinque *agent* e li riporta in un unico *log* testuale, come illustrato dall'estratto contenuto nel Listing 4.1. Questo file, oltre che ad avere utilità ai fini di *debug* (permettendo un controllo puntuale sui risultati di ogni singola frase), servirà poi a generare successivamente la *confusion matrix* (vedi Sezione 4.3) senza dover eseguire nuovamente il *testing*.

Lo *script* `test1.py` elabora anch'esso ognuno dei 21 file testuali, ma restituisce per ogni file –e quindi per ogni singolo *intent*– un valore normalizzato, secondo quanto definito nella funzione `getNormalizedSim` (algoritmo in pseudocodice nel Listing 4.2). La funzione calcola il valore di similarità per ogni frase del singolo file e somma tra di loro tutti i valori collegati all'*intent* bersaglio del file di test per poi normalizzarne il risultato sul numero totale di frasi presenti nel singolo file (ovvero 16). I risultati di ogni iterazione sono salvati su un file `.json` (ad es: `KFold(Test1-Training2345).result.json`) e vengono poi elaborati mediante lo *script* `getFinalResult.py` (algoritmo in pseudocodice nel Listing 4.3) che ne somma i risultati parziali e ne normalizza il totale secondo quanto visto alla Sezione 4.1 (i dati finali vengono scritti dallo *script* nel file `final.result.txt`, non riportato nel presente elaborato, dal quale sono estratti i dati riportati nel paragrafo successivo).

```
================================================
RESULTS FULL LOG OF THE K-FOLD CROSS VALIDATION
================================================
------------------------------------------------
RESULT FOR AGENT "KFold(Test1-Training2345)", INTENT "avere-
   ↪ caldo"
avere-caldo, score: 100%.<---Si muore di caldo
avere-caldo, score: 100%.<---Che afa!
avere-caldo, score: 100%.<---Ci saranno quaranta gradi fuori
(...)
------------------------------------------------
RESULT FOR AGENT "KFold(Test1-Training2345)", INTENT "insetto"
insetto, score: 100%.<---Oddio c'è un insetto
insetto, score: 100%.<---Un insetto!
(...)
```

Listing 4.1: Estratto dal file `fullLog.result.txt`, mostra il tipo di risultati fornito dallo *script* `test0.py`.

Risultati i dati estratti mediante lo *script* `getFinalResult.py` sono riportati nella Tabella 4.1 (nella Figura 4.2 è indicata come E_{TOT} *table*), con arrotondamento

```
input: file 'testFile.txt', string targetIntent
output: normalizedScore
begin
int totalSimilarity=0, simCounter=0
for line in 'testFile.txt'
      response ← getResponseFromDialogflow(ai, line)
      intent, score ← response
      if intent = targetIntent
            totalSimilarity ← score + totalSimilarity
      end if
      simCounter ← simCounter+1
end for
normalizedScore ← totalSimilarity/simCounter
```

Listing 4.2: Funzione `getNormalizedSim` estratta dal codice presente in `test1.py`

```
input: file 'KFold(Test1-Training2345).result.json',
            'KFold(Test2-Training1345).result.json',
            'KFold(Test3-Training1245).result.json',
            'KFold(Test4-Training1235).result.json',
            'KFold(Test5-Training1234).result.json'
begin
foreach result.json file
      create dictionary result
      result ← result.json file
end foreach
create dictionary finalResult
foreach result
      finalResult ← result + finalResult
end foreach
normalize finalResult (divide by 5)
totalAverage ← average overall the keys in finalResult
store finalResult, totalAverage in 'final.result.txt'
```

Listing 4.3: Pseudocodice per `getFinalResult.py`

alla terza cifra decimale[1].Per le stime delle singole iterazioni si è adottata la seguente

[1] L'arrotondamento è stato eseguito per eccesso nel caso di cifra 5 da eliminare preceduta da un numero dispari e per difetto nel caso di cifra 5 da eliminare preceduta da un numero

nomenclatura (in coerenza con quanto visto precedentemente per i nomi degli *agent*):

- E_1: si è utilizzato come *test set* il *fold* 1 e come *training set* i *fold* 2,3,4,5.
- E_2: si è utilizzato come *test set* il *fold* 2 e come *training set* i *fold* 1,3,4,5.
- E_3: si è utilizzato come *test set* il *fold* 3 e come *training set* i *fold* 1,2,4,5.
- E_4: si è utilizzato come *test set* il *fold* 4 e come *training set* i *fold* 1,2,3,5.
- E_5: si è utilizzato come *test set* il *fold* 5 e come *training set* i *fold* 1,2,3,4.

Come descritto nel Listing 4.3, i risultati di ogni iterazione sono stati normalizzati rispetto al singolo intento di partenza, in modo da avere 21 stime diverse (ognuna riportata in Tabella alla voce E), da mediare anch'esse per ricavare la stima totale (in Tabella indicata con E_{TOT}). Per verificare che non ci fossero fluttuazioni o contingenze particolari che inficiassero il risultato della validazione, l'algoritmo è stato poi iterato nella sua interezza per altre due volte. I dati di quest'ultimo passaggio sono presenti nella Tabella 4.2, dalla quale si può notare come il risultato della *cross validation* sia solido e non soggetto a particolari variazioni.

4.3 Confusion matrix

La *confusion matrix* [11, p.272] è un metodo di rappresentazione matriciale dei dati che evidenzia la correttezza delle predizioni di un determinato *model* mediante il confronto tra risultati attesi e risultati ottenuti. Nel caso attuale si tratta di una matrice 21x21 con aggiunta di una colonna ulteriore "D" dedicata al *Default Fallback Intent*, risultato non predetto ma ottenibile.

Grazie alla *confusion matrix* si può analizzare la tipologia di errori commessi, ed arrivare a conclusioni sul *model* anche più accurate che con la stima vista precedentemente, in modo da rafforzare i risultati già ottenuti. Contrariamente a quanto visto in precedenza, inoltre, l'analisi non è stata fatta prendendo in considerazione lo *score* (il punteggio che è stato interpretato come grado di similarità) ma solamente i risultati effettivi, vista poi l'importanza –a livello implementativo– non tanto del punteggio di similarità (che non viene restituito dal sistema) ma solamente della particolare situazione s_i identificata.

---

pari (arrotondamento gaussiano). Il calcolo di E è però stato eseguito senza arrotondare i dati, l'arrotondamento è posteriore ed a scopo di presentazione.

S	E_1	E_2	E_3	E_4	E_5	E
s_1	1.0	1.0	0.562	0.938	0.938	0,887
s_2	1.0	1.0	0.654	0.938	1.0	0,918
s_3	1.0	1.0	0.803	1.0	1.0	0,961
s_4	1.0	1.0	0.699	1.0	1.0	0,940
s_5	0.875	0.812	0.349	0.813	1.0	0,770
s_6	0.938	1.0	0.618	1.0	1.0	0,911
s_7	0.969	1.0	0.531	0.948	0.922	0,874
s_8	0.875	1.0	0.694	1.0	1.0	0,914
s_9	0.953	0.951	0.642	0.984	0.958	0,898
s_{10}	0.969	1.0	0.780	0.953	1.0	0,940
s_{11}	1.0	1.0	0.679	1.0	1.0	0,936
s_{12}	0.938	0.734	0.562	0.938	0.875	0,809
s_{13}	1.0	1.0	0.601	1.0	0.938	0,908
s_{14}	0.969	0.953	0.595	0.984	1.0	0,900
s_{15}	1.0	1.0	0.662	1.0	1.0	0,932
s_{16}	1.0	1.0	0.736	1.0	1.0	0,947
s_{17}	1.0	1.0	0.694	1.0	1.0	0,939
s_{18}	1.0	1.0	0.495	1.0	1.0	0,899
s_{19}	1.0	1.0	0.612	1.0	1.0	0,922
s_{20}	1.0	1.0	0.546	1.0	1.0	0,909
s_{21}	0.984	1.0	0.707	0.984	1.0	0,935
$\boldsymbol{E_{TOT}}$						**0,907**

Tabella 4.1: Risultati della 5-*fold cross validation* (calcolo №1).

№	E_{TOT}
1	0.907
2	0.901
3	0.901

Tabella 4.2: Risultati complessivi per i calcoli della 5-*fold cross validation*

	Output																					
S	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}	s_{11}	s_{12}	s_{13}	s_{14}	s_{15}	s_{16}	s_{17}	s_{18}	s_{19}	s_{20}	s_{21}	D
s_1	72	2	.	1	2	.	.	.	2	.	.	.	.	.	.	.	.	.	.	.	.	1
s_2	.	73	3	.	.	.	3	.	.	.	.	1	.	.	.	.	.	.	.	.	.	.
s_3	.	.	76	3	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1
s_4	.	.	.	77	1	.	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1
s_5	.	.	.	.	62	2	7	1	6	.	.	1	.	.	.	.	.	1	.	.	.	.
s_6	.	.	.	.	.	73	2	.	.	.	.	.	.	.	.	1	.	.	1	.	1	2
s_7	.	.	.	.	3	.	72	2	2	.	.	.	.	.	.	.	.	.	1	.	.	.
s_8	.	.	.	.	1	.	3	72	4	.	.	.	.	.	.	.	.	.	.	.	.	.
s_9	.	.	.	.	1	.	1	.	75	2	.	.	1	.	.	.	.	.	.	.	.	.
s_{10}	.	.	.	.	.	.	.	.	.	76	2	2	.	.	.	.	.	.	.	.	.	.
s_{11}	.	.	.	.	.	.	.	.	.	.	76	2	.	.	1	1	.	.	.	.	.	.
s_{12}	.	.	.	.	.	.	.	.	.	3	.	65	3	9	.	.	.	.	.	.	.	.
s_{13}	.	.	.	.	.	.	1	.	.	.	.	.	73	2	.	.	.	.	.	.	.	4
s_{14}	.	.	.	.	.	1	.	.	.	.	.	3	.	73	2	.	.	.	.	.	.	1
s_{15}	.	.	.	.	.	.	1	.	.	.	.	.	.	.	76	2	.	.	.	.	.	1
s_{16}	.	.	.	.	.	.	.	.	.	.	.	1	.	.	.	77	2	.	.	.	.	.
s_{17}	.	1	.	.	.	.	.	.	.	.	.	.	.	.	1	1	75	2	.	.	.	.
s_{18}	1	.	.	.	1	.	.	.	.	.	1	.	.	.	.	.	.	72	4	.	.	1
s_{19}	1	.	1	.	.	1	.	.	.	.	.	.	.	.	.	.	.	.	73	2	.	2
s_{20}	.	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	.	74	3	1
s_{21}	2	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	.	75	1

Tabella 4.3: Risultati della 5-*fold cross validation* (calcolo №1) nella *confusion matrix*

Per praticità di consultazione della Tabella 4.3 vengono riportate le corrispondenze s_i-situazione riprese dalla Tabella 2.3:

- $s_1 \triangleq$ Avere caldo
- $s_2 \triangleq$ Insetto nella stanza
- $s_3 \triangleq$ Piante fuori da bagnare
- $s_4 \triangleq$ Sapere che tempo ci sia fuori
- $s_5 \triangleq$ Aria da cambiare
- $s_6 \triangleq$ Rumore/stimolo da fuori
- $s_7 \triangleq$ Odore sgradevole
- $s_8 \triangleq$ Odore di gas
- $s_9 \triangleq$ Sentirsi soffocare
- $s_{10} \triangleq$ Avere sete
- $s_{11} \triangleq$ Andare a fare esercizio fisico
- $s_{12} \triangleq$ Mancanza di acqua a tavola
- $s_{13} \triangleq$ Fuoco da spegnere in casa
- $s_{14} \triangleq$ Mancanza di acqua corrente
- $s_{15} \triangleq$ Dover cucinare
- $s_{16} \triangleq$ Volersi informare sull'attualità
- $s_{17} \triangleq$ Voler guardare un film/serie tv
- $s_{18} \triangleq$ Aver bisogno di svagarsi/rilassarsi
- $s_{19} \triangleq$ Essere annoiati
- $s_{20} \triangleq$ Non riuscire a dormire
- $s_{21} \triangleq$ Troppo silenzio

La *confusion matrix* permette di eseguire interpolazioni di dati non solo tramite osservazione diretta, ma mediante alcuni parametri "classici" definiti in letteratura [11, p.272][14]. Si chiami V_i il numero di predizioni della situazione s_i corrette, F_i il numero di predizioni della situazione s_i errate ed N il numero totale di predizioni

eseguite:

$$\text{Accuratezza} = \frac{\sum_{i=1}^{21} V_i}{N}, \qquad \text{Tasso di errore} = \frac{\sum_{i=1}^{21} F_i}{N}$$
$$\text{Tasso di correttezza relativa} = \frac{V_i}{N}, \qquad \text{Tasso di errore relativo} = \frac{F_i}{N}$$

dai quali si può facilmente calcolare l'accuratezza ed il tasso di errore relativi al modello:

$$\text{Accuratezza} = \frac{\sum_{i=1}^{21} V_i}{N} = \frac{1537}{1680} = 0.9148$$
$$\text{Tasso di errore} = \frac{\sum_{i=1}^{21} F_i}{N} = \frac{143}{1680} = 0.0852$$

I dati relativi ai singoli intenti sono riportati, in valori percentuale, alla Tabella 4.4 che riproduce l'andamento della diagonale della *confusion matrix*. Come si è visto

Tasso di correttezza relativa ed accuratezza (%)									
s_1	90	s_6	91.25	s_{11}	95	s_{16}	96.25	s_{21}	93.85
s_2	91.25	s_7	90	s_{12}	81.25	s_{17}	93.75		
s_3	95	s_8	90	s_{13}	91.25	s_{18}	90		
s_4	96.25	s_9	93.75	s_{14}	92.25	s_{19}	91.25		
s_5	77.5	s_{10}	95	s_{15}	95	s_{20}	92.5	$\sum$:	**91.48**

Tabella 4.4: Accuratezza e tasso di correttezza relativa per singolo *intent*

quindi, il precedente risultato (90,7%, vedi Tabella 4.1) è stato confermato, ed anzi innalzato nel momento in cui si tralascia l'analisi degli *score* di similarità. Sebbene un'accuratezza del 91.5% ed un tasso di errore del 8.5% siano un buon risultato, dall'osservazione diretta della *confusion matrix* (Tabella 4.3) ci si può accorgere di alcuni risultati interessanti ottenuti grazie alla piattaforma. Essendo il *dataset* raccolto non filtrato rispetto agli input forniti, esso sarà un database rumoroso, ed è statisticamente probabile che contenga alcuni errori ed alcune imprecisioni di natura contingente dovute alla modalità di raccolta delle frasi. Gli interrogati a volte potrebbero aver fornito frasi troppo lontane dalla realtà o frasi più consone ad essere raccolte come indicatrici di un'altra situazione rispetto a quella in cui sono state poste. Inoltre tramite osservazione si può notare anche la qualità degli errori: se la *confusion matrix* riporta degli errori dovuti ale tra due situazioni simili (ad esempio "manca l'acqua a tavola"e "manca l'acqua corrente") questi sono da

considerarsi errori di una qualità maggiore perché dovuti ad una reale similarità tra le due situazioni, rispetto ad altre completamente diverse e senza porzioni semantiche in comune. Identifichiamo quindi due tipi di errore "di qualità": le correzioni al *dataset* e gli errori minori.

Correzioni al *dataset* Nella Tabella 4.5 sono riportati alcuni esempi di errori di riconoscimento avvenuti durante la 5-*fold cross validation*. Come si può notare, a causa del rumore presente nel dataset, le frasi utilizzate come *test phrase* hanno una correlazione molto bassa, nessuna in certi casi, con la situazione attesa, questo a causa dei già detti errori ed imprecisioni di compilazione da parte del gruppo di individui interrogati nei Form. È però da notare come i risultati errati siano molto più correlati (in alcuni casi quelli che in un database non rumoroso sarebbero i risultati attesi) che i risultati attesi stessi. Ad esempio, la *test phrase* "c'è odore di gas in cucina" ha sì una correlazione con la situazione attesa "aria da cambiare", ma sarebbe invece da inserire all'interno delle *training phrases* per la situazione "c'è odore di gas". Il *model* quindi ha dato una risposta errata, identificando come situazione per la frase "c'è odore di gas in cucina" la situazione "c'è odore di gas" invece che "aria da cambiare", il risultato però è corretto dal punto di vista operativo. Si può infatti immaginare che per confusione od imprecisioni la frase sia stata inserita nel Form in corrispondenza di una situazione ("aria da cambiare") che non le è congeniale, perlomeno rispetto ad altre (in questo caso, "c'è odore di gas"). Il *model* quindi, sbagliando, ha corretto l'errore presente nel *dataset*. Si è quindi potuto vedere come, anche con un *training set* limitato (ed ulteriormente limitato dal processo di validazione) si possa addestrare un *agent* in grado di correggere, in certi casi, gli errori presenti all'interno del suo *dataset*, filtrandolo dal rumore di inserimento.

Valore atteso	*Test phrase*	Output
s_5 = Aria da cambiare	C'è puzza di gas in cucina	s_8 = Odore di gas
s_5 = Aria da cambiare	C'è un cattivo odore	s_7 = Odore sgradevole
s_5 = Aria da cambiare	Non si respira	s_9 = Sentirsi soffocare
s_6 = Rum./stim. da fuori	C'è qualcuno in casa?	s_{21} = Troppo silenzio
s_8 = Odore di gas	Non respiro	s_9 = Sentirsi soffocare

Tabella 4.5: Esempi di correzioni al *dataset* estratti dal file `fullLog.result.txt` (vedi Figura 4.2).

Errori minori Come già accennato sopra, si sono identificati alcuni errori "di qualità", diversi dalle correzioni al *dataset*: gli errori minori. Rispetto agli errori visti precedentemente infatti, che possono essere interpretati come delle correzioni appunto, gli errori minori sono errori veri e propri, aventi però un'incidenza minore sull'abbassamento della qualità del modello. Come si può infatti vedere dagli esempi riportati alla Tabella 4.6, per questo tipo di errori il valore atteso condivide con il risultato del test una buona porzione semantica. Ad esempio, per la frase "manca l'acqua" il valore atteso "mancanza di acqua a tavola" e quello registrato "mancanza di acqua corrente" sono estremamente simili, è quindi parzialmente giustificata la confusione. Altre volte invece la *test phrase* utilizzata è ambigua e non è strettamente legata ad una situazione piuttosto che ad un'altra, come nel caso della frase "apri la finestra!" che, in quanto non identificante una situazione ma un comando, non è correlata strettamente ad una sola situazione.

Valore atteso	*Test phrase*	Output
s_1 = Avere caldo	Non si respira	s_5 = Aria da cambiare
s_5 = Aria da cambiare	C'è aria viziata in camera	s_7 = Odore sgradevole
s_5 = Aria da cambiare	Apri la finestra!	s_7 = Odore sgradevole
s_{12} = Manc. acq. tavola	Manca l'acqua	s_{14} = Manc. a. corrente
s_{12} = Manc. acq. tavola	Non c'è più acqua	s_{14} = Manc. a. corrente

Tabella 4.6: Esempi di errori minori estratti dal file `fullLog.result.txt` (vedi Figura 4.2).

4.4 Necessità di un'analisi approfondita

Al *testing* mediante *k-cross fold validation* è stato affiancato un *testing* effettuato su utenti che manualmente interagissero col sistema realizzato. Durante questa interazione diretta è stata evidenziata una problematica. Determinate frasi inserite come input ("sudo un sacco", "sudare un sacco"...) non venivano identificate correttamente (s_1). Era infatti restituito in uscita il *default fallback intent*. Vista l'evidente correlazione tra frase testata e situazione aspettata (s_1) si sono provate alcune varianti, tutte identificate correttamente ("sudo terribilmente"- *score*: 49%, "sto sudando un sacco"- *score*: 77%). Non solo, la stessa frase iniziale se isolata poteva portare a una corretta identificazione ("sudo"- *score*: 49%, "sudare"- *score*: 49%). Questo risultato ha portato alla luce la necessità di un'analisi ulteriore per i

casi come questo, in modo da poter evitare il *default fallback* come risultato forzando il riconoscimento della frase mediante un passaggio successivo.

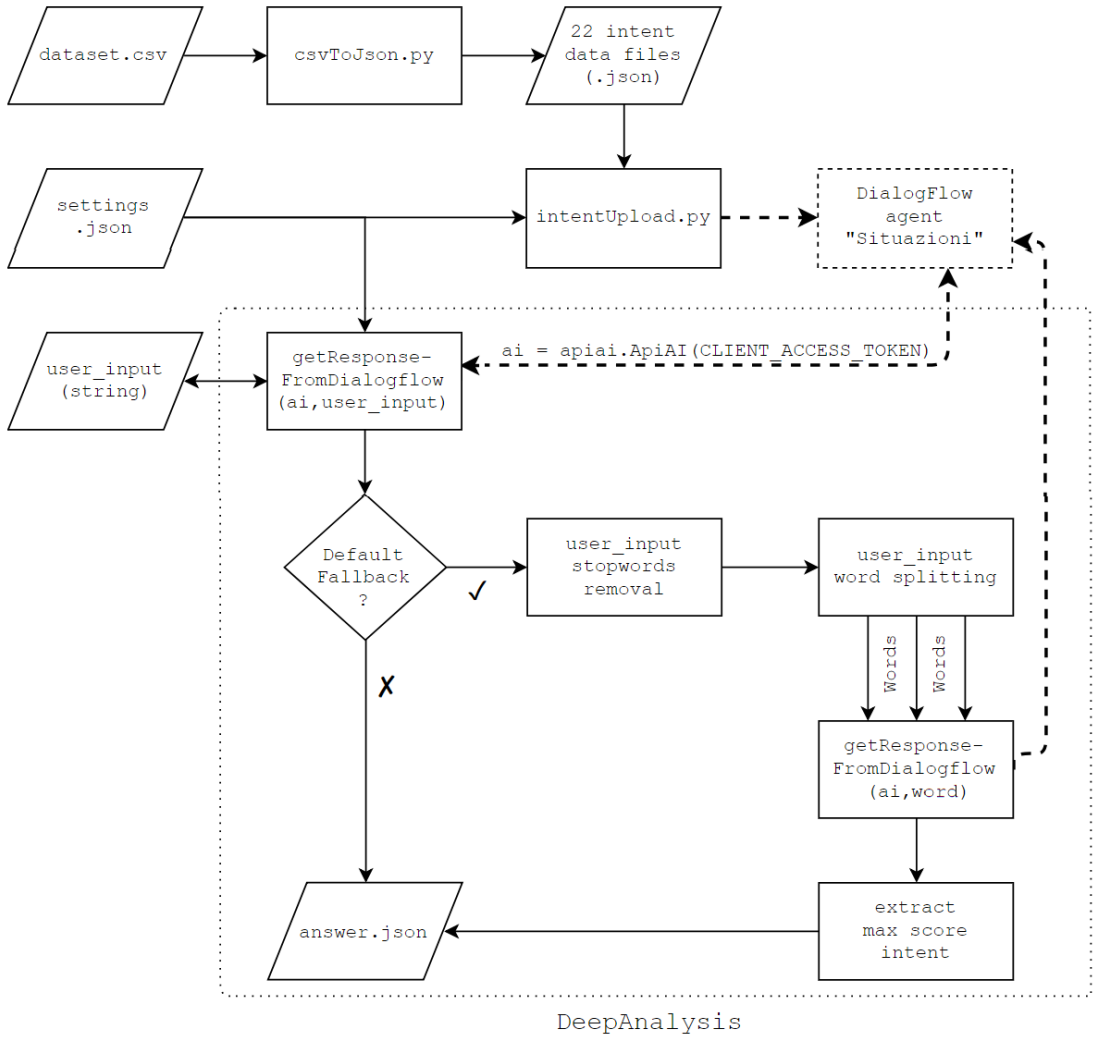


Figura 4.3: Schema generale del processo di addestramento e di identificazione mediante analisi ulteriore.

Deep analysis Come illustra la Figura 4.3, rispetto al processo di identificazione già visto bisogna aggiungere una parte ad attivazione condizionale, che processi ulteriormente lo `user_input` nel caso di risposta *default fallback*. Per fare ciò il client `client.py` iniziale (visto in Figura 3.3 e nei paragrafi dedicati) è stato

modificato per implementare il processo di analisi approfondita. Il tutto è quindi stato implementato in un client alternativo `deepAnalysis.py`, che riassume i processi visti alla Figura 4.3 (pseudocodice nel Listing 4.4). L'analisi ulteriore consiste nel processare singolarmente le parole contenute nell'input, in modo tale da ottenere informazioni aggiuntive mediante esse ed evitare problemi quali quello illustrato precedentemente. Come ogni processazione semantica sulle singole parole di una stringa in linguaggio naturale, sarà però doveroso un filtraggio dell'input, tipico delle applicazioni di *Natural Language Processing* [4][9]. Il processo di eliminazione delle *stopword* (le parole comuni che vengono utilizzate in un linguaggio con grande frequenza e quindi non legate a particolari contesti semantici [9]) è necessario per evitare di avere risposte completamente casuali dovute ad una distribuzione non uniforme delle *stopword* all'interno del *dataset*. A tal proposito vengono sfruttate due librerie Python (il *Natural Language Toolkit* `ntlk` e la libreria `stopwords`) per importare due liste predefinite di *stopword* per la lingua italiana e, mediante la lista, lo `user_input` viene filtrato in modo da mantenere solamente le parole con una specificità semantica più marcata. L'input viene poi suddiviso (*splitted*) nelle singole parole, le quali vengono individualmente processate dall'*agent* DialogFlow e poi confrontate tra loro. Per quanto riguarda i possibili algoritmi di confronto, si sono avanzate diverse proposte (vedi Sezione 7.2) ma per l'implementazione qui vista si è utilizzato come criterio di selezione lo *score* della singola parola: l'*intent* identificato sarà quello avente il punteggio di similarità più alto (vedi Listing 4.4 e Figura 4.3).

Conseguenze sull'accuratezza Per quanto riguarda l'accuratezza del *model* successivamente a questa modifica, si può affermare con certezza –per le modalità di implementazione nel sistema precedentemente testato– che l'accuratezza sia sicuramente maggiore o uguale a quella precedentemente calcolata, andando il sistema ad agire solamente sulle risposte di *default fallback*, quindi sicuramente errate, con la possibilità di variarne alcune in meglio (le variazioni in peggio sono comunque errori, anche se il punto meriterebbe un approfondimento ulteriore che qui si è omesso per brevità). Essendosi quindi rilevati 16 risposte di *fallback* in fase di validazione (vedi Tabella 4.3), l'accuratezza può variare fino a quasi un punto percentuale in più, un risultato importante se si considera la semplicità di implementazione dell'analisi approfondita e l'accuratezza già alta ottenuta. Per una panoramica delle possibili integrazioni tra risultati dell'algoritmo precedente e risultati della *deep analysis*

```
input: string 'user_input', file 'settings.json'
output: string intent, file 'answer.json'
begin
tokens ← 'settings.json'
ai ← createDialogFlowSession(tokens)
response ← getResponseFromDialogflow(ai, user_input)
intent ← response
if intent is 'Default Fallback Intent'
      import stopwords from packages 'ntlk','stopwords'
      user_input ← user_input - stopwords
      words ← split(user_input)
      for word in words
            respOfWord ← getResponseFromDialogflow(ai, word)
            intentOfWord,scoreOfWord ← response
      end for
      intent ← intentOfWord with max scoreOfWord
      response ← respOfWord with max scoreOfWord
end if
store response in 'answer.json'
return intent
```

Listing 4.4: Pseudocodice per `deepAnalysis.py` (vedi Figura 4.3)

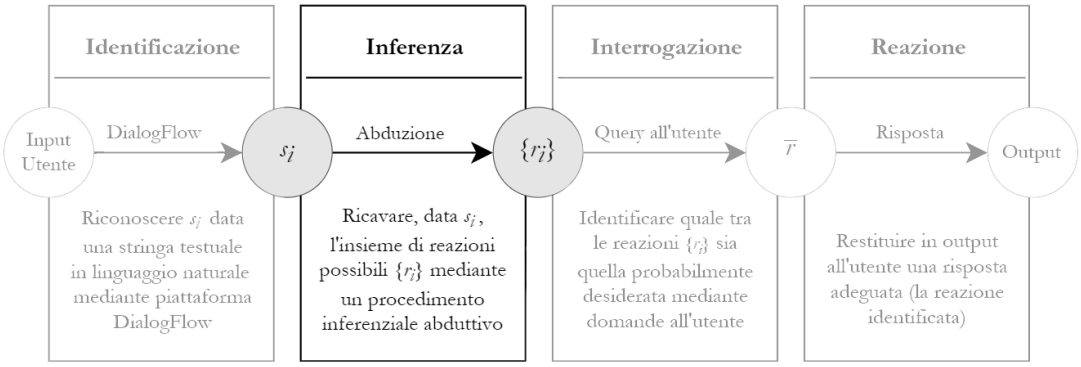
diverse da quella più semplice qui mostrata, si rimanda alla Sezione 7.2.

Capitolo 5

Inferenze ed interrogazioni

Dopo aver progettato la parte di sistema dedicata all'identificazione della situazione, ed avendo analizzato l'accuratezza del modello proposto, in questo Capitolo si metterà a punto il sistema di reazione alla situazione, nelle sue tre fasi di inferenza, interrogazione e reazione vera e propria.

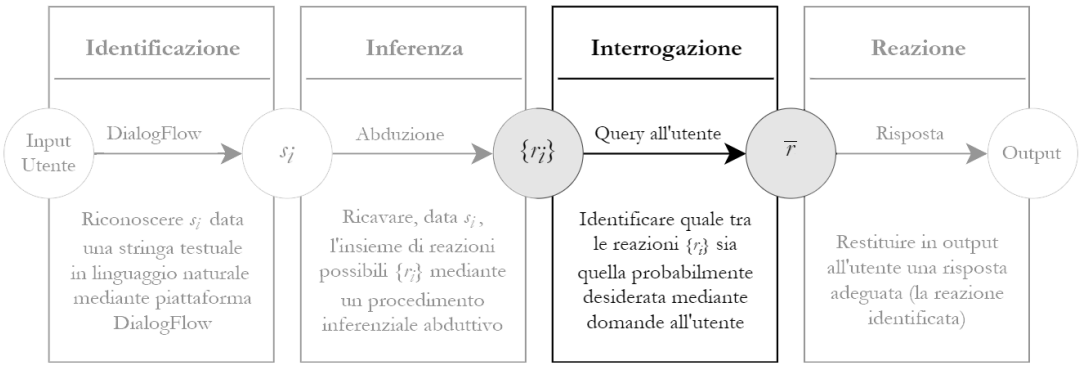
5.1 Inferenza



Una volta restituita la $s_i \in S$ da DialogFlow, il sistema deve identificare mediante un procedimento inferenziale le possibili reazioni corrisponenti. Il procedimento, modellato a partire dai meccanismi del ragionamento abduttivo tenendo a mente l'approccio convenzionale alla pragmatistica, è stato tradotto con una tabella situazioni-probabilità, una *look-up table* si potrebbe dire, dipendente da un contesto (come la Tabella 2.2 o la Tabella 2.3) che associa ad ogni situazione la probabilità per ogni singola reazione $r_i \in R$. Dal punto di vista dell'inferenza, però, non

è richiesta la probabilità (basta una tabella di KB come quella vista alla Tabella 2.1). La probabilità associata alle singole reazioni è determinante infatti in fase di interrogazione. Per semplicità si è gestito assieme il meccanismo di inferenza ed interrogazione, dando la tabella di probabilità la possibilità di ricavare sia le possibili $\{r_i\}$, sia le corrispondenti probabilità (vedi Sezione successiva). Concettualmente, però, la fase di inferenza è separata da quella di interrogazione, ed ha il compito di isolare abduttivamente le possibili ipotesi $H = r_i$ dall'insieme degli assunti A (vedi Sezione 1.3 e Capitolo 2).

5.2 Interrogazione



Il processo di inferenza e di interrogazione, come visto nella Sezione 2, segue quello di inferenza delle possibili reazioni. Dato infatti l'insieme di reazioni $\{r_i\}$ ricavate dalla tabella contestuale situazioni-probabilità (nel caso qui proposto è la Tabella 2.3) il sistema progettato deve identificare la reazione $r_i \in R$ adatta alla situazione identificata. Si è quindi pensato di stabilire una "soglia di sicurezza" CT (*certainty threshold*) al di sotto della quale il sistema non è in grado di reagire con sicurezza ed ha quindi bisogno di interrogare l'ambiente circostante (l'utente, nell'esempio qui proposto). Definiamo quindi:

$$r_i \quad \text{sicura } (certain) \Leftrightarrow P(r_i) > CT$$

Nel caso di tabelle contestuali situazioni-probabilità più estese, con più reazioni in comune, la definizione della CT è una problematica da non trascurare. Nel caso del *toy model* analizzato in questa sede, però, avendo una tabella limitata e non molte reazioni comuni a singole situazioni si è scelto di utilizzare come soglia $CT = 0\%$ (sono sicure le reazioni r_i t.c. $P(r_i) > 0$).

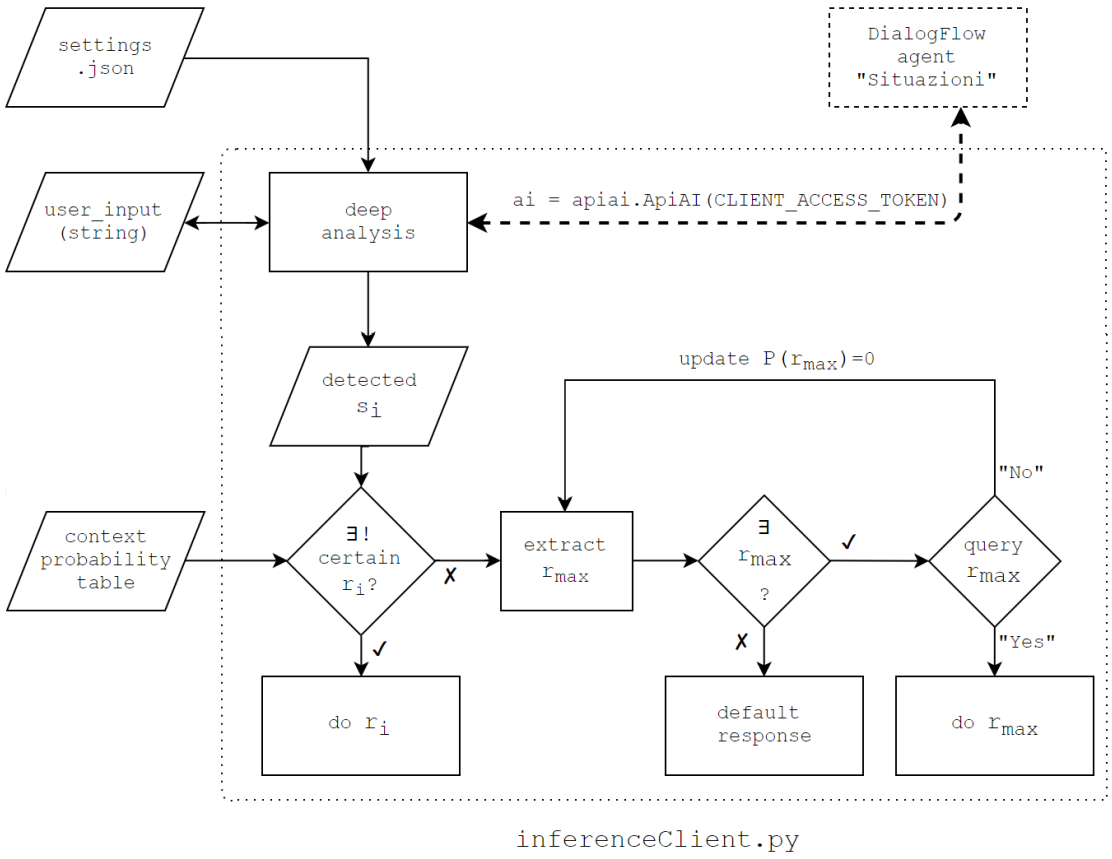


Figura 5.1: Schema generale del processo di identificazione e successiva inferenza e interrogazione mediante analisi ulteriore.

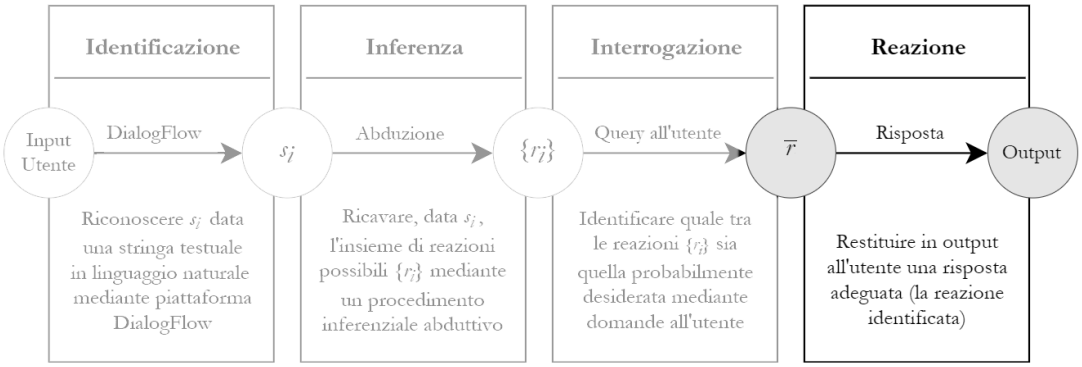
Si è scelto di implementare il meccanismo di inferenza e interrogazione all'interno dello stesso *script* utilizzato precedentemente per l'identificazione (espandendo lo script `deepAnalysis.py` nello *script* `inferenceClient.py`, come è possibile vedere nella Figura 5.1). Dopo aver identificato la situazione $s_i \in S$ il sistema confronta i valori di probabilità delle reazioni $r_i \in R$ corrispondenti e, se ne identifica una ed una sola "sicura" ($\exists!\ r \in R\ t.c.\ P(r) > CT$), allora procede all'esecuzione della reazione. Altrimenti ($\neg\exists!\ r \vee \nexists\ r\ t.c.\ P(r) > CT$) vi sono altre due possibilità:

- se non esiste alcuna r_i compatibile ($P(r_i) = 0, \forall r_i \in R$) verrà restituita una risposta di default del tipo "Mi dispiace, non penso di poter fare molto per te"(vedi Figura 5.1)
- se esistono più r_i compatibili il sistema inizierà ad interrogare l'utente allo

scopo di capire se sia desiderata la r_i con probabilità più alta (`query` r_{max}) ed a risposta negativa azzererà temporaneamente la probabilità della reazione passando a quella successiva, avente quindi il valore maggiore tra tutte (vedi sempre Figura 5.1).

Nel caso di molte r_i è possibile inserire una ulteriore soglia di certezza sotto la quale il sistema non pone domande in caso di disambiguazione tra troppe r_i (eliminando quindi le possibilità lontanamente probabili). Questa è però un'opzione da adottarsi solo per tabelle contestuali situazione-probabilità molto estese, e non è peculiare del caso qui presentato.

5.3 Reazione



Come già anticipato, il sistema qui progettato è limitato all'interazione testuale con l'utente ed alla estrazione di dati e direttive a partire da enunciati del linguaggio naturale. Non sono perciò tenute in considerazione tematiche quali la modellazione fisica o le caratteristiche delle architetture robotiche ospiti. La fase di reazione quindi, una volta identificata $\bar{r}$, consiste semplicemente nel restituire in riposta all'utente una frase del tipo "Penso eseguirò l'azione $\langle \bar{r} \rangle$".

Il sistema può quindi ora dirsi completo: a dato input vocale è in grado di identificare la possibile situazione comunicata dall'utente (significato esplicito) e mediante una tabella contestuale ricavare abduttivamente una reazione probabilmente desiderata tra quelle definite nel modello, essendo in grado di disambiguare tra reazioni possibili tramite interrogazione diretta ed interazione con l'utente.

Capitolo 6

Analisi complessiva

Nel presente Capitolo verrà affrontata l'analisi del sistema completo, basandosi su quanto già detto riguardo all'analisi di efficacia al Capitolo 4, integrando i risultati già visti per l'identificazione con un *testing* via utente complessivo che metta alla prova nella pratica il sistema realizzato.

6.1 Accuratezza complessiva

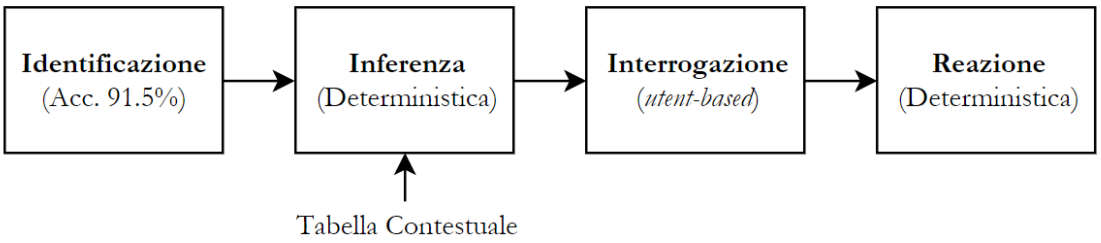


Figura 6.1: Diverse fasi del processo e rispettive caratteristiche in funzione agli errori.

Identificazione Come illustrato alla Figura 6.1, le diverse fasi del processo presentano differenze evidenti e fondamentali, dovute alle differenze di progettazione dei vari sotto-sistemi realizzati nel corso del lavoro. Per quanto riguarda la fase di identificazione, questa è stata realizzata mediante una piattaforma, DialogFlow, che sfrutta tecniche di *machine learning*. Come visto esaustivamente nel Capitolo 4

queste tecniche necessitano di una validazione dei risultati, in modo da ricavare un punteggio di efficacia con il quale valutare il *model* predittivo realizzato. Si è visto che l'accuratezza del sistema di identificazione è del 91,5%, un buon risultato se visto assieme alle ulteriori considerazioni sugli errori "di qualità" precedentemente trattati. Si chiami quindi α_s l'accuratezza di questo sotto-sistema.

Inferenza Il processo di inferenza, come visto nella Figura 6.1, è invece deterministico. Il risultato quindi, data una determinata s_i, è sempre e comunque l'insieme corretto di $\{r_i\}$, dove per "corretto" si intende un risultato in coerenza con la tabella contestuale fornita. L'accuratezza del sotto-sistema è quindi del 100%, assumendo che la Tabella contestuale fornisca dati realistici. Si chiami quindi α_{t1} l'accuratezza di questo sotto-sistema, dipendente in toto dalla tabella contestuale. Si è quindi supposto $\alpha_{t1} = 100\%$, un valore realistico vista la modalità di raccolta dati mediante il primo Form, essendo questo il metodo migliore per raccogliere dati attendibili riguardanti comportamenti comuni della quotidianità.

Interrogazione Similmente all'inferenza, il processo di interrogazione è deterministico e si basa –oltre che sull'insieme $\{r_i\}$– sui valori forniti dalla Tabella contestuale. Diversamente dal precedente, però, è soggetto anche a dinamiche complesse e non deterministiche: essendo *user-based* il risultato del processo dipende dall'interazione con l'utente e dalle risposte che questi dà al sistema. Si suppone però che le risposte dell'utente siano accurate e coerenti, e se anche ciò non fosse risposte errate non inficiano la qualità del sistema in assoluto. Come visto perciò per la fase precedente, anche per questo sistema esiste un'accuratezza α_{t2}, dipendente in toto dalle probabilità fornite dalla tabella contestuale. Si chiami quindi α_{tab} l'accuratezza $\alpha_{t1} \cdot \alpha_{t2}$ della tabella contestuale, sia per quanto riguarda le associazioni situazione-reazione sia per quanto riguarda le corrispettive probabilità.

Reazione La fase di reazione vera e propria è stata quella approfondita di meno nell'elaborato, vista la natura *software* del sistema qui presentato. Il processo di reazione è quindi puramente deterministico, dovuto alle associazioni tra r_i e risposta i-esima in output.

L'accuratezza totale del sistema, quindi, dipende da due fattori principali: l'efficacia dell'identificazione e la qualità della tabella contestuale (supposta $\stackrel{?}{=} 100\%$):

$$\alpha = \alpha_s \cdot \alpha_{tab} \stackrel{?}{=} 91.5\%$$

Si vedrà nel seguito se l'accuratezza ricavata mediante mezzi "teorici", per così dire, verrà confermata da test reali su utenti interagenti con il sistema.

6.2 Testing su utenti

La fase di *testing* finale su utente ha lo scopo di confermare empiricamente i dati trovati mediante la validazione e l'analisi di efficacia precedentemente illustrata. Verranno create alcune situazioni vincolate in cui gli utenti, opportunamente istruiti, interagiranno con il sistema per testarne un ampio spettro di possibilità. Ad ogni individuo interrogato viene dato un set di istruzioni riguardanti le possibili azioni che egli può ottenere dal robot (prendere una bottiglia d'acqua, accendere la televisione, aprire la finestra), un insieme composto da sette situazioni $S^* = \{s_i\}_{i\in[1,7]} \subset S$ ed alcuni vincoli di natura formale. L'utente dovrà quindi, per ogni $s_i \in S^*$, descrivere la propria situazione con lo scopo di far eseguire al robot uno dei tre comandi a piacere. Come interazione di esempio viene fornita agli utenti la frase "c'è la torta che sta bruciando"come possibile modo per comunicare il comando "spegni il forno". I vincoli da rispettare sono:

- non nominare le parole contenute in una lista di "parole vietate" $V = \{$accendere, aprire, prendere, portare, acqua, bottiglia, finestra, televisione$\}$,
- non inserire in input enunciazioni che siano comandi diretti (fare/portare/eseguire...) come "spegni il forno". Le enunciazioni ammesse sono infatti descrittive della particolare situazione, perlomeno dal punto di vista puramente letterale ("c'è la torta che sta bruciando").

Ogni utente ha cinque *run* per ogni situazione, in modo da testare uno spettro più ampio di possibili input collegati alle situazioni. Verranno analizzati due fattori:

- α = rapporto tra *run* corretti e *run* totali,
- β = numero di *run* medio richiesto per ottenere il primo esito positivo.

Risultati I risultati del testing sono riportati nella Tabella 6.1, con riferimento ai punteggi complessivi per singola situazione s_i. Il risultato finale, sebbene ottenuto mediante un bacino ristretto di utenti, permette una iniziale contro-validazione della tabella di probabilità. Si è ricavato infatti

$$\alpha = \alpha_s \cdot \alpha_{tab} = 82\%$$

un risultato differente rispetto all'accuratezza del 91.5% ricavata "teoricamente". Questo ci permette di correggere l'accuratezza della tabella:

$$\alpha_{tab} = \frac{\alpha}{\alpha_s} = \frac{82\%}{91.5\%} = 89.6\%$$

Il numero ristretto di utenti utilizzati per la raccolta dati e per il *testing* (dell'ordine di grandezza delle decine), dovuto alla metodologia di raccolta e di verifica, ha sicuramente inficiato l'accuratezza del sistema, fornendo una tabella non precisa. Estendere il Primo Form a popolazioni di interrogati più ampie, arrivando a numeri dell'ordine delle centinaia, è sicuramente la soluzione più intuitiva, ma estremamente dispendiosa se si pensa allo sforzo necessario rispetto alla semplicità del *toy model* qui presentato. La modalità del *testing* inoltre, forzata da una interazione innaturale con un computer invece che ad una interazione in prima persona con il robot in un contesto coerente, ha spinto gli interrogati a ricercare input non del tutto naturali, espressi in forme inusuali o forzate (il valore β evidenzia bene come, per le prime interazioni, immediate, naturali e meno forzate, le prestazioni siano ottime essendo il risultato si prossimo al valore minimo di 1). Al Capitolo 7 verranno illustrate alcune proposte per eliminare queste restrizioni, intuizioni di interesse che potrebbero, se correttamente sviluppate, portare le potenzialità del sistema studiato al di fuori del *toy model* qui affrontato.

S			α	β
s_1	$\triangleq$	Avere caldo	90%	1
s_2	$\triangleq$	Insetto nella stanza	86%	1
s_3	$\triangleq$	Piante fuori da bagnare	86%	1
s_4	$\triangleq$	Sapere che tempo ci sia fuori	84%	1
s_5	$\triangleq$	Aria da cambiare	90%	1
s_6	$\triangleq$	Rumore/stimolo da fuori	70%	1
s_7	$\triangleq$	Odore sgradevole	82%	1
s_8	$\triangleq$	Odore di gas	86%	1
s_9	$\triangleq$	Sentirsi soffocare	82%	1,5
s_{10}	$\triangleq$	Avere sete	84%	1
s_{11}	$\triangleq$	Andare a fare esercizio fisico	84%	1
s_{12}	$\triangleq$	Mancanza di acqua a tavola	82%	1
s_{13}	$\triangleq$	Fuoco da spegnere in casa	82%	1,5
s_{14}	$\triangleq$	Mancanza di acqua corrente	82%	1
s_{15}	$\triangleq$	Dover cucinare	74%	1
s_{16}	$\triangleq$	Volersi informare sull'attualità	82%	1
s_{17}	$\triangleq$	Voler guardare un film/serie tv	84%	1
s_{18}	$\triangleq$	Aver bisogno di svagarsi/rilassarsi	70%	1
s_{19}	$\triangleq$	Essere annoiati	86%	1
s_{20}	$\triangleq$	Non riuscire a dormire	70%	1
s_{21}	$\triangleq$	Troppo silenzio	86%	1
TOT			**82%**	**1.05**

Tabella 6.1: Risultati dell'analisi su utenti effettuata.

Capitolo 7

Ulteriori sviluppi e conclusioni

Vengono qui delineate, prima delle conclusioni, alcune prospettive di evoluzione del sistema e proposte per studi ulteriori al fine di rendere i robot implementanti le funzionalità qui trattate competitivi in ambienti operativi più estesi e complessi del *toy model* presentato. Le suddette prospettive sono state pensate sia per rimediare ad alcuni errori ed imprecisioni notati durante lo studio qui affrontato (ad esempio migliorare l'indice di accuratezza generale) sia per aggirare i limiti del modello semplificato, il quale ha imposto restrizioni molto forti su alcuni parametri che operativamente, in un ambiente dinamico e complesso, non permettono un corretto funzionamento del sistema.

7.1 Tavola di probabilità e *dataset*

La tavola contestuale situazioni-probabilità è stata considerata fissa, con valori definiti rispetto ai risultati del Primo Form (Appendice A). Un successivo miglioramento sarebbe eliminare i due problemi principali di questo approccio: la dipendenza da un contesto fisso ed immutabile e la pesantezza della raccolta dei dati per tabelle più estese.

Tavola adattiva Per quanto riguarda il primo vincolo (la dipendenza da un contesto fisso), creare delle tabelle che siano dinamiche e varino temporalmente a seconda del contesto che si registra sarebbe probabilmente la soluzione migliore e meno dispendiosa computazionalmente. Tale meccanismo sarebbe implementabile, ad esempio, tramite un secondo livello di identificazione per contesti più generali,

mediante i quali adattare le tavole situazioni-probabilità caso per caso. Il contesto potrebbe essere identificato tramite input vocali diretti oppure, più efficacemente ma con un'implementazione ulteriore, mediante stimoli ambientali, come illustrato nella Figura 7.1. È infatti ovvio notare come le convenzioni linguistiche mutino a seconda dei contesti (come accennato alla Sezione 1.4) e come una identificazione preventiva del contesto possa evitare malintesi dovuti ad abduzioni basate su convenzioni non applicabili al contesto del proferimento.

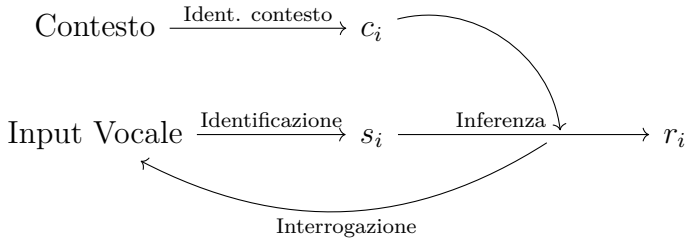


Figura 7.1: Schema logico di funzionamento del sistema con tavola contestuale adattiva (si rimanda alla Figura 2.2 per il confronto)

***Dataset* generato automaticamente** Per quanto riguarda il secondo vincolo (la pesantezza del raccolta dei dati per tabelle più estese) è chiaramente un problema molto più grande e complesso del primo, ma aggirabile creando degli algoritmi che in contesti più ampi, quelli di reale applicazione del sistema, costruiscano in maniera automatica un *dataset* con rispettiva tavola contestuale consistente. La costruzione automatica del *dataset* potrebbe avvenire a partire da un insieme minimale, da espandere mediante regole e procedimenti mirati a catturarne le possibili varianti, senza dover interrogare attivamente mediante Form o strumenti simili. Inoltre, come visto nel caso qui trattato, la costruzione del *dataset* ha un margine di errore accettabile, vista la capacità di correzione del sistema per insiemi rumorosi. Partendo da un certo *dataset* si potrebbero applicare regole e procedimenti di espansione, ad esempio espandendo il *corpus* iniziale con regole basate su grammatiche formali che, dato un numero di frasi ristrette e basandosi su un dizionario di sinonimi, permettano di ottenere un gran numero di varianti sintattiche delle frasi di partenza, ad esse semanticamente correlate (si veda, ad esempio lo sviluppo delle grammatiche formali della lingua inglese in [10, Cap. 11]). La tabella di probabilità sarebbe quindi quella di partenza –bisognerebbe fornire quindi subito tutte le possibili s_i

contemplate dall'applicazione– ma con un numero di esempi per situazione molto minore (se progettate correttamente le regole di espansione). Il fatto però di avere un insieme predefinito ed immutabile S da costruire preventivamente è comunque una limitazione, che potrebbe essere aggirata mediante sistemi che tentino di "apprendere" come identificare nuove situazioni. Nel seguito è riportata (con i suoi aspetti problematici) una proposta in tal senso.

Apprendimento cumulativo Un'altra possibile soluzione al problema sarebbe un sistema senza alcuna tabella contestuale iniziale, oppure con una tabella minimale *general-purpose*, capace di apprendere mediante semplici comandi. Il sistema dovrebbe essere in grado di inserire nuove situazioni, qualora istruito dall'utente, ed associare le nuove frasi non riconosciute ad esse o a situazioni preesistenti, mediante la creazione di *intent* e la modifica delle singole *training phrase* degli *intent corrispondenti* in DialogFlow. Come illustrato alla Figura 7.2 il funzionamento è identico a quanto già visto alla Figura 2.2 (in Figura vengono omessi con (...) i passi logici del processo proprio perché già visti) ad eccezione del comportamento in seguito ad una risposta che non identifica la situazione. In questo caso infatti viene implementata una procedura ulteriore che, mediante l'interazione con l'utente, determina se aggiungere la frase non identificata come *training phrase* di un determinato *intent* oppure creare un nuovo *intent* corrispondente alla situazione $s \notin S$ appena identificata ed aggiungere l'input vocale quale prima *training phrase* per l'*intent* corrispondente (nella Figura 7.2 l'interrogazione è rappresentata in 'Query').

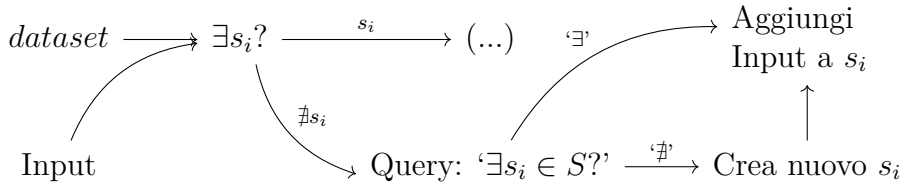


Figura 7.2: Schema logico di funzionamento del sistema ad apprendimento cumulativo

Un sistema di questo tipo, nonostante la curva di apprendimento non veloce, potrebbe essere la soluzione adatta per la diffusione di un robot così programmato in ambiti molto diversi da loro o non previsti. Si potrebbero anche creare delle banche dati personalizzate mediante la condivisione dei *dataset* dei singoli sistemi

impiegati in campi simili, in modo da fornire successivamente una base consolidata sul quale esercitare la propria personalizzazione mediante interazione diretta.

Per creare una tavola di probabilità per un sistema del genere, del quale non si possono prevedere gli sviluppi e fornire quindi le probabilità preventivamente, si può adottare un calcolo del seguente tipo:

$$P(s_i \Rightarrow r_i) = \frac{\text{Nº } (s_i \Rightarrow r_i) \text{ registrati}}{\text{Nº } (s_i) \text{ identificate}}$$

Ogniqualvolta quindi una reazione r_i viene inferita ed eseguita, viene aggiornata la rispettiva probabilità (questa quindi subisce un aumento). Nel caso la r_i inferita non sia però quella corretta, e ciò sia notificato al sistema dall'utente, la probabilità viene aggiornata ma come se l'inferenza non fosse avvenuta (la probabilità diminuisce). Lo schema logico generale di un sistema ad apprendimento automatico sarebbe quindi del tipo illustrato in Figura 7.3.

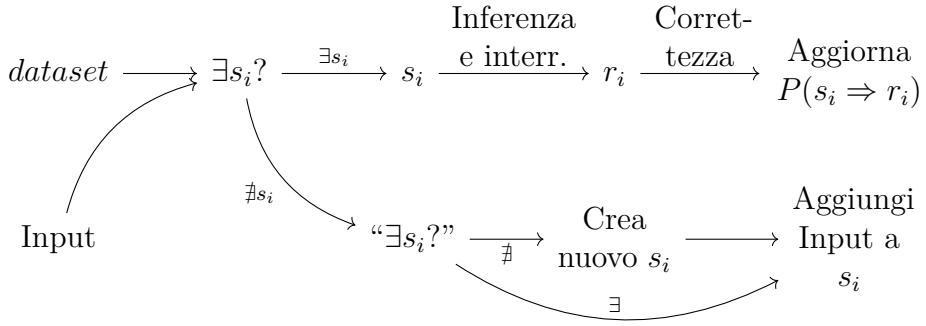


Figura 7.3: Schema logico complessivo del sistema ad apprendimento

È da notare come in un sistema ad apprendimento descritto in questi termini l'insieme R, essendo l'insieme delle reazioni eseguibili dal robot, non è modificabile dall'utente mediante interazioni, essendo le funzionalità di esso intuitivamente fissate in partenza dalle sue possibilità fisiche e dalle capacità che gli sono state conferite in fase di progettazione fisica. Ciò non toglie che sia possibile, in seguito ad una modifica funzionale, mappare le nuove r_i con un insieme di s_i preesistenti (con qualche domanda iniziale all'utente) e poi procedere come visto.

Un possibile modo per non cristallizzare la dinamica del sistema dopo diverso tempo (vista la difficoltà di modifiche sostanziali alle probabilità avendo grandi numeri a denominatore) sarebbe quella di conservare un numero h di casi (*history* del sistema) tale per cui una interazione viene "dimenticata" nel momento in cui è succeduta da un numero h di interazioni in cui è stata identificata la stessa s_i.

È immediato notare come le probabilità rispetto all'insieme R di una situazione s appena appresa non siano definite. Le soluzioni in questo caso potrebbero essere, ad esempio, il definire una probabilità di *default* per ogni $r_i \in R$ rispetto ad s (approccio semplicistico facile da gestire ma non molto efficace) o facendo interagire il robot con l'utente domandando, dopo l'apprendimento, quali possibili reazioni sarebbero da associare. Si potrebbe anche inserire un sistema del genere in seguito all'analisi di correttezza della situazione (vedi Figura 7.3). Se, come si è detto, prima che il robot esegua la reazione l'utente ne blocca l'esecuzione con un comando diretto (l'utente esegue un analisi di correttezza dichiarando errata la reazione al suo proferimento) questi può essere interrogato dal robot sul tipo di reazione che avrebbe voluto vedere eseguita, aggiornando poi (chiedendo prima conferma all'utente del collegamento effettivo tra situazione e reazione) le rispettive probabilità. La questione solleva però problemi di natura logica ed implementativa che non è possibile trattare in questa sede, ma che sarebbe interessante approfondire successivamente.

7.2 Algoritmi di identificazione

Come già anticipato alle Sezioni 3.4 e 4.4, l'algoritmo di identificazione utilizzato nello studio qui presentato non è l'unico possibile. Sono infatti state pensate diverse varianti e combinazioni, equivalenti o di complessità superiore al tipo di algoritmo di analisi utilizzato nel sistema presentato. Le possibili proposte per gli algoritmi sono qui presentati suddivise per numero di output: prima sono presentate quelle più semplici da gestire –a singolo output– poi quelle ad output multipli.

7.2.1 Algoritmi a singolo output

Gli algoritmi a singolo output restituiscono, in base all'ingresso dato dall'utente, una situazione s_i corrispondente (oppure un *default*). L'algoritmo di analisi approfondita presentato nell'elaborato, ad esempio, è di questo tipo.

Algoritmo 1 *(Standard) Sia dato l'input testuale immesso dall'utente.*
– *L'*intent *restituito in output è quello identificato dall'*agent *DialogFlow.*

Algoritmo 2 *(Deep analysis) Sia dato l'input testuale immesso dall'utente.*
– *L'input viene filtrato dalle* stopword *e poi suddiviso nelle singole parole.*
– *Viene identificato dall'*agent *DialogFlow un* intent *per ogni parola.*
– *L'*intent *restituito in output è quello con maggiore* score *tra quelli identificati.*

Algoritmo 3 *(Standard esteso) Sia dato l'input testuale.*
– *Viene eseguito l'Algoritmo 1.*
– *In caso di* default *viene eseguito l'Algoritmo 2.*

L'Algoritmo 3 (Standard esteso) è quello utilizzato nella precedente trattazione (dopo l'implementazione del meccanismo di analisi ulteriore).

Algoritmo 4 *(Deep analysis cumulativa) Sia dato l'input testuale.*
– *L'input viene filtrato dalle* stopword *e poi suddiviso nelle singole parole.*
– *Viene identificato dall'*agent *DialogFlow un* intent *per ogni parola.*
– *Per ogni* intent *si calcola lo* score *totale sommando quelli delle singole parole.*
– *L'*intent *restituito in output è quello con maggiore* score *totale.*

Algoritmo 5 *(Deep analysis cumulativa normalizzata) Sia dato l'input testuale.*
– *L'input viene filtrato dalle* stopword *e poi suddiviso nelle singole parole.*
– *Viene identificato dall'*agent *DialogFlow un* intent *per ogni parola.*
– *Per ogni* intent *si calcola lo* score *totale sommando quelli delle singole parole.*
– *Ogni* score *totale viene normalizzato rispetto al numero di parole di quell'*intent
– *L'*intent *restituito in output è quello con maggiore* score *totale normalizzato.*

Algoritmo 6 *(Deep analysis estesa) Sia dato l'input testuale.*
– *Vengono eseguiti l'Algoritmo 2, l'Algoritmo 4 e l'Algoritmo 5.*
– *Viene quindi restituito in output l'*intent *in output comune ad almeno due degli Algoritmi eseguiti, altrimenti viene restituita una risposta di* default.

Diversamente dall'Algoritmo di *deep analysis* utilizzato nell'elaborato, l'Algoritmo 4 e l'Algoritmo 5 pongono attenzione non tanto alla singola parola quanto alle parole nel loro insieme, ciascuno con una modalità diversa. L'Algoritmo 6 media tra questi tre approcci, fornendo il risultato forse più bilanciato. Qualsiasi considerazione da farsi sull'accuratezza di questi algoritmi è però del tutto inutile senza una adeguata analisi di efficacia come quella vista per l'Algoritmo 3.

Inferenza ed interrogazioni Per il processo di inferenza ed interrogazione si attua il procedimento visto nel presente lavoro: data una situazione s_i identificata, si controlla la probabilità di inferenza corrispondente $P(s_i \Rightarrow r_i)$; se è poi una reazione r_i sicura ($P(s_i \Rightarrow r_i) > CT$) si procede all'attuazione della stessa, altrimenti si attua un interrogazione al fine di disambiguare tra le possibilità.

7.2.2 Algoritmi ad output multipli

Algoritmo 7 *(Deep analysis estesa con incertezza) Sia dato l'input testuale*
– Vengono eseguiti l'Algoritmo 2, l'Algoritmo 4 e l'Algoritmo 5.
*– Viene restituito in output l'*intent *in output comune ad almeno due degli Algoritmi eseguiti, altrimenti viene restituita una risposta "incerta" in cui vengono riportati tutti e tre gli* intent *identificati.*

Algoritmo 8 *(Deep analysis cumulativa multivalore) Sia dato l'input testuale.*
– L'input viene filtrato dalle stopword *e poi suddiviso nelle singole parole.*
*– Viene identificato dall'*agent *DialogFlow un* intent *per ogni parola.*
– Per ogni intent *si calcola lo* score *totale sommando quelli delle singole parole.*
– Gli intent *restituiti in output, tra quelli identificati, sono quelli con* score *totale maggiore della soglia CT.*

Algoritmo 9 *(Deep analysis cumul. normal. multivalore) Sia dato l'input testuale.*
– L'input viene filtrato dalle stopword *e poi suddiviso nelle singole parole.*
*– Viene identificato dall'*agent *DialogFlow un* intent *per ogni parola.*
– Per ogni intent *si calcola lo* score *totale sommando quelli delle singole parole.*
– Ogni score *totale viene normalizzato rispetto al numero di parole di quell'*intent
– Gli intent *restituiti in output, tra quelli identificati, sono quelli con* score *totale normalizzata maggiore della soglia CT.*

Inferenza ed interrogazioni Per il processo di inferenza ed interrogazione vi sono diversi modi per gestire una risposta multipla. Nel caso dell'Algoritmo 7 ci si comporta esattamente come per gli algoritmi a singolo output, tranne che nel caso di risposta incerta. Un possibile modo per risolvere i casi multipli con diverse $s_i \in S$ restituite sarebbe quello di sommare le rispettive probabilità di inferenza corrispondenti $P(s_i \Rightarrow r_i)$; se poi emerge una sola reazione r_i sicura ($P(s_i \Rightarrow r_i) > CT$) si procede all'attuazione della stessa, altrimenti si attua un interrogazione al fine di disambiguare tra le possibilità.

È da notare come per gli algoritmi ad output multipli, in particolare per gli ultimi due visti (l'output incerto dell'Algoritmo 7 potrebbe essere interpretato come una correzione aggiuntiva all'Algoritmo 6 a singolo output), sia estremamente difficile capire quanto effettivamente possa riscontrarsi un miglioramento, senza affrontare un'analisi di efficacia. Per come è stato concepito il sistema, però, questo tipo di algoritmi non sembra aiutare molto, in quanto non incontra esattamente le esigenze

del modello logico iniziale, perlomeno non quanto gli altri possibili sviluppi prima presentati.

Gli algoritmi qui presentati sono solo alcuni dei possibili, e le possibili combinazioni tra essi non sono state che accennate. Lo scopo di tali proposte è quindi di fornire alcuni possibili elementi di partenza per analisi di efficacia ulteriori, volte a capire quali di questi procedimenti proposti e quale combinazione di essi sia la più efficace ed accurata, anche in sinergia con le possibili proposte illustrate alla Sezione 7.1.

7.3 Conclusioni

L'obiettivo di questo studio era quello di analizzare le potenzialità del procedimento abduttivo in un sistema robotico interagente con un utente umano. Mediante l'abduzione il robot doveva mostrarsi in grado di estrarre, mediante informazioni contestuali, i significati implicati dai proferimenti dipendenti dal contesto (indicali, implicature, atti linguistici...). Lo studio si proponeva quindi di analizzare l'efficacia di un approccio convenzionale alla decodifica di queste forme linguistiche, in modo che il robot potesse reagire correttamente a proferimenti di queste tipologie. Il *toy model* definito ha dato ottimi risultati in seguito all'analisi di efficacia ed i risultati ottenuti confermano la validità dell'approccio scelto. Su questa base, sono stati inoltre proposti alcuni possibili sviluppi ulteriori, allo scopo di superare le limitazioni poste in fase di analisi.

Essendo la problematica estremamente vasta ed il fenomeno complesso e dalle numerose sfaccettature, non si ha qui la presunzione di affermare di aver progettato un sistema in grado di reagire correttamente ai proferimenti dipendenti dal contesto. Tuttavia si è mostrato come, soprattutto in ambienti ristretti, l'approccio convenzionale –unito ad una modellazione probabilistica delle ipotesi ricavate dal procedimento abduttivo– porti a risultati importanti. Le proposte per superare i limiti del modello qui presentato sono da leggersi in tal senso, quale un possibile passo successivo per avvicinarsi ad una modellazione coerente del fenomeno nella sua interezza.

Appendice A

Risultati Primo Form

Prima Domanda: Immagina di essere in casa tua e di voler aprire la finestra. Per quali possibili cause potresti volerlo fare?

S	%
Caldo	10 %
Aria da cambiare	10 %
Insetto nella stanza	7 %
Stendere	7 %
Rumore o altro stimolo proveniente dalla strada	5 %
Odore sgradevole	4 %
Piante fuori da bagnare	4 %
Lavare i vetri della finestra	4 %
Aprire le persiane	4 %
Salutare/parlare con una persona fuori	4 %
Guardare che tempo fa	3 %
Chiudere le persiane	3 %
Andare sul balcone/in giardino	3 %
Odore di gas	2 %
Parlare con un vicino di casa	2 %
Illuminare la stanza	2 %
Guardare il panorama	2 %
Sentirsi soffocare	2 %
Dover sbattere un tappeto/dei panni	2 %
Fumare	2 %
Per fare una foto	2 %
Fumo nella stanza	1 %
Pavimento bagnato	1 %
Prendere una boccata d'aria	1 %
Ritirare il bucato	1 %

Parlare al telefono	1 %
Per lanciare un oggetto a qualcuno	1 %
Piccioni sul cornicione	1 %
Umidità	1 %
Vernice delle pareti fresca	1 %
Musica/suono/odore piacevole proveniente da fuori	1 %
Suona il citofono	1 %
Far entrare direttamente il sole	1 %
Prendere il sole	1 %
Guardare l'arcobaleno	1 %
Disturbare i vicini con il rumore della propria casa	1 %
Sentire la temperatura	1 %
Sbattere la tovaglia	1 %
Guardare la pioggia	1 %
Per far vedere ai vicini che si è tornati	1 %
Mettere a raffreddare qualcosa fuori	1 %
Passare a qualcuno fuori un oggetto	1 %

Tabella A.1: Risultati del primo Form - Prima Domanda

Seconda Domanda: Immagina di essere in casa tua e di voler prendere una bottiglia d'acqua. Per quali possibili cause potresti volerlo fare?

S	%
Avere sete	12 %
Annaffiare dele piante	8 %
Offrirla a degli ospiti/altre persone	7 %
Doversela portare dietro quando si esce	6 %
Dover andare a fare esercizio fisico	5 %
Mancanza di acqua in tavola	4 %
Un fuoco libero in casa	4 %
Sono finite le bottiglie d'acqua in frigo	4 %
Mancanza di acqua corrente in casa	4 %
Buttarla via perchè è vuota	4 %
Mancanza di acqua nella macchinetta del caffè	3 %
Dover prendere un medicinale	3 %
Serve per cucinare	3 %
Avere caldo	2 %
Voler fare una tisana/the/bibite	2 %
Mancanza di acqua nel ferro da stiro	2 %
Un animale domestico ha sete	2 %

Per schiacciare un insetto	2 %
Per fare esercizi con le braccia	2 %
Per riempirla d'acqua	2 %
Voler pulire qualcosa	2 %
Riempire una borraccia/travasarla in un altra bottiglia	1 %
Bagnare qualcuno/fargli uno scherzo	1 %
Tirarla addosso a qualcuno	1 %
Metterla in degli stampini per ghiaccio	1 %
Avere la bocca secca	1 %
Stare sudando	1 %
Aver mangiato del cibo troppo asciutto	1 %
Avere liquido che non si sa dove mettere	1 %
Avere un cattivo gusto in bocca	1 %
Essere disidratati	1 %
Schiarirsi la voce	1 %
Aprirla	1 %
Fermare una porta se c'è corrente	1 %
Diluire un concentrato	1 %
Tagliarla ed usarla come contenitore	1 %
Smacchiare un abito	1 %
Lavare la macchina	1 %
Leggerne l'etichetta	1 %
Metterla vicino al letto in vista della notte	1 %
Sciacquare una ferita	1 %
Controllare che non perda	1 %
Svuotarla	1 %
Capire se è naturale o frizzante	1 %
Riempire una piscina	1 %
Per il pretesto di dovermi alzare	1 %
Per darla vuota al cane per farcelo giocare	1 %
Per sciacquare un bicchiere	1 %

Tabella A.2: Risultati del primo Form - Seconda Domanda

Terza Domanda: Immagina di essere in casa tua e di voler accendere la televisione. Per quali possibili cause potresti volerlo fare?

S	%
Doversi informarmi sull'attualità	14 %
Voler guardare un film/serie tv	10 %
Doversi svagare (es: videogiochi)	9 %
Aver Noia	7 %
Voler vedere un programma specifico	7 %
Voler ascoltare della musica	7 %
Aver bisogno di rilassarmi	7 %
Sentirsi soli	5 %
Aver saputo che sarà trasmesso qualcosa che interessa	4 %
Non riuscire a dormire	3 %
Controllare che il tutto sia funzionante	3 %
Voler sapere il meteo	3 %
Dover coprire un rumore molesto/gente che urla	3 %
Voler vedere video registrati	3 %
Troppo silenzio	2 %
Voler un sottofondo	2 %
Voler imparare qualcosa	1 %
Voler infastidire i vicini	1 %
Volere sfogarsi	1 %
Non doversi addormentare	1 %
Fare zapping cercando programmi interessanti	1 %
Voler intrattenere gli ospiti	1 %
Voler ascoltarla mentre si cucina	1 %
Voler leggere il televideo	1 %
Volere risintonizzarla	1 %
Volere registrare un programma	1 %
Guardare qualcosa in compagnia	1 %
Vi è una richiesta da parte di qualcuno	1 %
Voler distrarre qualcuno	1 %
Voler sapere l'ora	1 %

Tabella A.3: Risultati del primo Form - Terza Domanda

Appendice B

Risultati Secondo Form

Nel seguito si elenca per completezza il *corpus* di frasi ricavato dalle risposte date al secondo Form:

1. Immagina di essere in casa e di avere caldo:

Che caldo!
Si muore di caldo.
Che caldo!
Mamma ho caldo.
Che caldo!
Che caldo!
Fa troppo caldo in questa stanza.
Ho caldo.
Che caldo.
Che caldo!
Ho caldo.
Si muore di caldo.
Si muore di caldo.
Ho caldo.
Ho caldo.
Fa molto caldo.
Che caldo!
Sto sudando.
Che afa.
Fa caldissimo!
Papà fa molto caldo.
Sto morendo di caldo.
C'è molto calore.
Oggi fa molto caldo.
La temperatura oggi è troppo alta.
Che afa.
Mi sto sciogliendo.
La temperatura è alta.
Sembra di essere ai tropici.
C'è afa.
Che afa.
Sto svenendo dal caldo.
Percepisco una temperatura elevata.
Oggi si muore!
Non si respira con questo caldo.
Ci saranno quaranta gradi fuori.
Troppo caldo, oggi!
Ceci ho caldo.
Si soffoca!
Oggi la temperatura è molto alta.
Solo io ho caldo?
Sto sudando eppure sono qui seduto.
M*****a è un forno qua dentro.
Sto sudando troppo.
Mi sembra di essere nel deserto.
Sto sudando come un porco.
Non si respira.
Si muore qui!
Sto sudando.
Questa stanza è torrida.
Non si respira!
C'è afa.
Quest'estate è particolarmente torrida.
Senti che caldo?
Zia ho caldo.
Ci vorrebbe un po' d'aria.
Questo caldo mi sfianca.
Sto morendo di caldo.
Il calore qui oggi è insopportabile.
Mi sto sciogliendo qua.
Ho bisogno di una doccia.
Questa stanza sembra un forno.
Sto facendo una sauna.
Sto sudando.
È un forno sta stanza.
Si soffoca dal caldo.
Mi sento accaldato.
Che afa!
Ogni fa troppo caldo.
Il sole oggi è incandescente.
Che caldo, quanti gradi saranno?
Mimmo fa caldo.
Sembra di essere nel deserto.
Oggi non respiro dal caldo.
Fa troppo caldo, mi sto sciogliendo.
Non senti anche tu il caldo di questa giornata?
Fa troppo caldo qua.
È proprio arrivata l'estate.
Sto sudando moltissimo.
Non c'è un filo d'aria.
Sto cuocendo.
Ci saranno 500 gradi qui dentro.
Quest' afa mi uccide.
La stanza è calda.
Grondo sudore!

2. Immagina di essere in casa e di vedere un insetto nella stanza:

che schifo, un insetto!
Oddio c'è un insetto.
Oddio, c'è un insetto!
Papà c'è un insetto enorme nella stanza.
Che schifo, un insetto!
C'è un insetto in camera.
Guarda un ragno!
È appena entrata una vespa.
Cos'è quello?
C'è un insetto gigante!
C'è un insetto.
È entrato un insetto.
Oddio c'è un animale!
Che schifo una bestia alata.
Alfred, c'è un insetto.
Vedo un insetto.
Da dove è entrato?
Aiutami a far uscire questo insetto.
Aiuto c'è una cimice sul muro.
C'è un insetto!
Oddio mamma c'è un insetto.
C'è un insetto sul

muro.
Che schifo, è entrato un insetto.
Occhio che c'è una bestia che gira.
Hai un calabrone sulla spalla.
Un insetto!
Aprite le finestre che c'è un insetto.
Sento un ronzio.
Sento un ronzio.
C'è un insetto che mi ronza intorno!
Un insetto aiuto.
Che schifo, un insetto.
Un insetto è entrato in casa.
Servirebbe qualcosa per schiacciarlo.
Prendi un giornale, è entrato un insetto.
Sono terrorizzata dall'ape che è appena entrata.
Aspetta, c'è un insetto!
C'è un insetto dove sono io.
Vado a prendere l'ammazzamosche.
Chissà da dove è entrato l'insetto.
Vado a prendere l'insetticida perché è entrato un insetto.
Questa mosca che mi ronza nelle orecchie è insopportabile.
Che è quel coso?
Dicono porti fortuna quell'insetto.
Qualcosa mi ha punto.
Odio le zanzare.
C'è un insetto sul divano!
Ci mancava che entrasse un insetto.
C'è un insetto fastidioso che mi gira intorno.
Che orrore, un insetto!
Forse dovrei metterlo in salvo.
Oddio è entrato un insetto.
È entrata una bestia in casa.
Che schifo, cos'è quel coso?
Mamma c'è un insetto qui.
È entrato un insetto.
Voglio uscire, perché c'è un insetto.
Ho paura sia entrata una bestia dalla finestra.
Questa zanzara che gira attorno mi ha appena punto.
È un insetto quello?
Attenti a non calpestare l'insetto.
Odio gli insetti.
Potrei essere punto.
C'è un insetto che vola!
Oddio che è quel robo con le zampe.
Alfred, quell'oliva si muove.
C'è un insetto.
Che schifo!
Apri la finestra che c'è un insetto.
Prendi la paletta, è entrata una mosca.
Uffa c'è un insetto!
Aiuto c'è un insetto.
Cos'è quel coso che vola?
Che spavento, c'è un insetto!
C'è una brutta bestia che cammina sul muro.
Nella stanza c'è un calabrone.
Un bestio!
Ma che insetto é quello?
Aiuto un insetto.
Ci sarà un nido di vespe qui fuori.
C'è una bestiaccia!
Che schifo odio gli insetti.
C'è un insetto, alfred passami un giornale.
Sbaglio o quello è un insetto?
Ci mancava solo questa!

3. Immagina di essere in casa e di notare che le piante fuori siano da bagnare:

mi passi l'annaffiatoio.
Le piante mi sembrano un po' secche.
Le piante sono secche.
Mamma bisogna annaffiare le piante, la terra è secca.
Vanno annaffiate le piante.
Le piante hanno sete.
Secondo te le piante hanno bisogno di essere bagnate?
Le piante del terrazzo hanno sete.
Uh le piante sono secche.
Quelle piante sono secche.
Le piante stanno seccando.
Le piante sembrano secche.
Ci sono le piante che hanno sete.
Povere piantine sembrano sciupate.
Alfred, bisogna dare dell'acqua alle piante.
Credo che le piante necessitino l?acqua.
Sembrano un po' giù.
Non dimentichiamoci di bagnare le piante.
Annaffi tu le piante o lo faccio io?
Dobbiamo dar da bere alle piante.
Farò bene ad annaffiare quelle piante.
Le piante stanno morendo.
Penso che le piante abbiano bisogno d'acqua.
Mi sa che mi sono dimenticata di bagnare le piante ieri.
Bisogna che diamo dell'acqua alle piante.
Stanno seccando le piante fuori?
È ancora presto per bagnare le piante.
Speriamo piova.
Se non inizia a piovere, dovremo bagnare le piante.
Bisogna bagnare le piante.
È un po' che non piove, le piante saranno assetate.
Le piante stanno soffrendo.
Spero che le piante non siano seccate.
Non sarebbe opportuno bagnarle?
Ancora un po' e saranno da buttare.
Ci sono le piante da bagnare.
Hai già bagnato le piante?
Bisogna bagnare le piante.
È meglio annaffiare le piante secche.
Bisogna dare l'acqua alle piante.
Vedo che le piante sono da bagnare.
Pensi sia un errore bagnare le piante ora che c'è ancora il sole?
La terra dei vasi fuori è troppo asciutta.
Cavolo vanno bagnate le piante.
Le piante hanno sete.
L'anno prossimo compro solo piante grasse che sono meno impegnative.
Se non bagniamo subito le piante, moriranno.
Le piante stanno patendo il caldo.
La terra delle piante sembra un po' secca.
Se non diamo dell'acqua alle piante, moriranno.
Vedo le piante fuori disidratate.
Forse dovrei fare qualcosa.
La terra di quelle piante è secca.
Dov'è l'innaffiatoio, è ora di annaffiare le piante.
Le piante non hanno acqua.
Devo annaffiare le piante assolutamente.
La terra delle piante è secca.
Bisogna annaffiare le piante.
Forse dovrei bagnare le piante.
Bisogna innaffiare I geranei.
Devo bagnare le piante.
É stata una giornata afosa oggi devo bagnare le piante.
La terra dei vasi è arida.
Con questo caldo le piante hanno bisogno di molta acqua.
La terra delle piante è secca.
Mi son dimenticata di innaffiare I fiori.
Alfred, le piante mi sembrano secche.
A meno che non piova, bisognerà bagnare le piante.
Non sembrano molto in forma.
Quelle piante hanno sete.
Bisogna bagnare le piante.
Dobbiamo dare un po' d'acqua alle piante.
C'è bisogno di annaffiare le piante fuori.
Mi serve l'annaffiatoio.
Nessuno ha annaffiato le piante.
Mi daresti una mano a bagnare le piante in giardino?
Le piante stanno appassendo per la mancanza d'acqua.
Sbaglio o devo bagnare le piante?
Stasera bagnerò le piante.
L'impianto di irrigazione si è rotto.
Oh no, ieri sera ho scordato di bagnare le piante!
Le piante hanno bisogno di un po' d'acqua.
Sarà meglio che prenda l'annaffiatoio.
Alfred, le piante sono da bagnare.
Dai da bere alle piante.
È un po' che non piove.

4. Immagina di essere in casa e di non sapere quale sia il tempo fuori di casa :

com'è fuori?
Com'è la temperatura fuori?
Che tempo fa oggi?
Mamma che tempo fa?
Com'è il tempo oggi?
Che tempo fa fuori?
Fa caldo fuori?
Chissà se ora sta piovendo.
Com'è il tempo fuori?
Speriamo ci sia bel tempo fuori.
L'aria è umida, chissà se sta piovendo.
Che tempo c'è fuori?
Chissà che tempo c'è fuori.
Chissà che tempo fa fuori.
Chissà che tempo fa fuori.
Che tempo farà fuori?
Chissà come è oggi?
Speriamo ci sia bel tempo oggi.
Non so come sia il clima fuori.
C'è il sole fuori?
Com'è il cielo?
C'è il sole oggi?
Secondo te oggi è nuvoloso o sereno?
Com'è il tempo fuori?
Hai mica visto se è nuvoloso ?
Fuori piove?
In casa fa fresco, chissà fuori.
Chissà se fuori fa freddo come in casa.
Sono stato chiuso qui dentro così a lungo che non so più che tempo faccia fuori.
Ci sarà vento fuori?
Ultimamente ha fatto bello, chissà se regge.
Sarà uscito il sole fuori?
Spero piova, dato che è tanto che non lo fa. Ma non ne sono certo.
Potrò fare una lavatrice?
Hai visto le previsioni del tempo?
Com'è il tempo?
Quanti gradi ci sono fuori?
Com'è il tempo?
Come si sta fuori?
Non so che tempo faccia fuori.
Sta ancora piovendo o ha smesso?
Non ho ancora capito che tempo fa oggi.
Fuori c'è il sole?
Chissà se servirà una giacchetta stasera.
Per sicurezza ho sempre un ombrello in borsa.
Pioverà o ci sarà il sole?
Speriamo che fuori non piova.
Vorrei sapere come vestirmi oggi.
Alfred, sta piovendo?
Non so quale sia il tempo fuori: spero sia bello.
Che tempo farà?
Oggi dovrebbe piovere?
C'è il sole?
Com'è il tempo fuori?
Mamma puoi dirmi com'è la giornata fuori.
Piove?
Puoi dirmi com'è il tempo?
Come è il tempo oggi?
Non ho visto se fuori c'è il sole.
Com'è il meteo?
Spero non piova,ho fatto il bucato.
Non so se portarmi una felpa oppure no.
Speriamo che ci sia il sole.
Sarà bello fuori?
Non si capisce se fa bello o se fa brutto.
Alfred, come è il tempo?
Bisognerebbe consultare un meteo per sapere il tempo fuori, o affacciarsi a una finestra.
Speriamo sia bello.
Secondo te farà caldo fuori?
Secondo te fa caldo fuori?
Fa freddo fuori?
Mamma è bel tempo?
Fa freddo fuori?
Devo controllare il tempo fuori.
Fa freddo fuori o si sta bene anche senza giacca?
È sereno o coperto stamattina?
Com'è fuori?
Se c'è bel tempo potremmo uscire.
La statua segna tempo è diventata rosa.
Dal rumore sembra ci sia una bufera.
Chissà se c'è freddo fuori.
Potrò uscire senza ombrello oggi?
Alfred, come è il cielo?
Dovrei affacciarmi per vedere il tempo che fa.
Sarà bello?

5. Immagina di essere in casa e che nella stanza in cui ti trovi l'aria sia da: cambiare:

che puzza in questa stanza.
In questa stanza l'aria è viziata.
Apriamo un po' le finestre!
Bisogna far circolare l'aria nella stanza.
C'è odore di chiuso.
Che aria pesante.
In questa stanza manca l'aria.
C'è aria viziata.
Soffoco.
C'è aria viziata qui.
C'è poco ossigeno nella stanza.
C'è odore di chiuso.
C'è puzza.
Qui si soffoca.
Non si respira qua dentro.
Che puzza di chiuso.
Apriamo le finestre.
Che odore di chiuso.
C'è odore di chiuso.
C'è puzza nella stanza.
C'è aria viziata, apriamo le finestre?
Va cambiata l'aria nella stanza.
Bisogna arieggiare.
Qui l'aria è da cambiare.
C'è odore di chiuso.
Non c'è ossigeno.
Non si respira qua.
Fa un po' caldo in questa stanza.
C'è odore di chiuso.
Anche se fa freddo apro la finestra, bisogna cambiare aria.
C'è aria viziata in camera.
Qua va cambiata un po' l'aria.
Che odore di chiuso.
In questa stanza non si respira.
Un po' di aria nuova non guasterebbe.
Non si respira qui.
Non si respira in questa stanza.
Apri la finestra!
L'aria in questa stanza deve essere assolutamente cambiata.
Che puzza.
Bisogna cambiare l'aria.
Non si respira.
Non c'è aria in questa stanza.
Cambiare un po' l'aria?
Facciamo uscire la mosca.
Le finestre sono sigillate?
Sembra di essere in una stalla.
Non si respira in salotto.
Che aria pesante che c'è qui dentro.
Mi manca l'aria alfred.
Cambiamo l?aria, c'è puzza di morto.
Che odore!
Fai girare un po' l'aria qui dentro.
Bisogna cambiare l'aria.
Cambiamo un po' l'aria!
L'aria in questa stanza puzza, bisogna farla circolare.
Non si respira.
Non hai ancora aperto la finestra?
C'è l'aria da cambiare.
L'aria è pesante.
Manca aria anche a te?
Ossigeno?
C'è puzza.
L'aria è irrespirabile.
La mattina è sempre meglio cambiare aria.
Che cattivo odore in sala.
Che puzza di chiuso.
C'è da cambiare l'aria alfred.
Bisognerebbe cambiare l?aria qui.
L'aria è stagnante.
Senti che puzza che c'è qui.
L'aria è un po' rarefatta.
Dobbiamo cambiare l'aria!
Sento che l'aria va cambiata.
Vado ad aprire la finestra.
Non riesco a respirare bene.
C'è un cattivo odore.
Non si respira qui.
Apriamo una finestra?
Vorrei fumare una sigaretta.
Che cattivo odore.
In questa stanza non si respira.
C'è puzza di gas in

cucina.
Sarà meglio aprire una finestra.
Non c'è ossigeno qui.
C'è un?aria pesantissima,
bisogna cambiarla.
Non si può respirare.

6. Immagina di essere in casa e che da fuori provenga un rumore o uno stimolo insolito:

ho sentito un rumore.
Ho sentito un rumore.
Cos'è stato?
Mamma cosa è stato?
Cos'è stato?
Senti anche tu questo rumore?
Ho sentito un rumore.
Avete sentito che botto?
Cos'era quel suono?
Speriamo non sia successo nulla di strano.
Caspita che botta!
Cos'è stato?
Cos'è stato quel rumore?
Cos'è stato quel rumore?
Che strano rumore.
Cos?è stato?
Mai sentito questo rumore.
Viene uno strano rumore da fuori.
Chi sta urlando nel vicinato?

Hai sentito quel suono?
Che succede?
Che rumore è?
Secondo te chi sta facendo questo baccano?
Ho sentito uno strano rumore.
Assordante questo suono improvviso!
Uh?
Che rumore fastidioso.
Mi sono spaventata.
Sbaglio, o sembrava un colpo di pistola?
C'è qualcuno in casa?
Cosa starà facendo tutto sto casino.
Che fastidio questo rumore.
Hai sentito? Che strano rumore.
Che fanno I vicini?
Lo hai sentito anche tu quel rumore?
Cos'è questo frastuono?
Cos'era quel rumore?

Che strano rumore, cos'è?
Cosa stanno facendo fuori?
Sento un rumore strano.
Credo che fuori stia succedendo qualcosa.
Cosa è stato questo rumore?
Avete sentito anche voi?
Non ho sentito cosa hai detto per via del rumore.
L'allarme dei vicini è fastidioso.
Cosa può essere a quest'ora?
C'è stato un rumore inquietante!
Ma che combinano I vicini.
Hai sentito anche tu, alfred?
C'è qualcosa di insolito fuori.
Staranno facendo dei lavori?
Cos'era quel rumore?

C'è caos proveniente dall'esterno.
Hai sentito?
Cos'è questo rumore?
Cos'è questo rumore!
Che cos'è tutta questa confusione?
Mi è parso di sentire un rumore venire da fuori.
Che spavento. Non me lo aspettavo.
Che succede fuori?
Anche oggi devono fare I lavori!
Ci sarà stato un incidente?
Mi è sembrato di sentire uno strano rumore.
È caduto qualcosa?
Che succede là fuori?
Cos'è è stato, alfred?
Ho sentito un rumore, cosa sarà?
Cosa sarà a fare questo rumore?
Oddio che spavento quel rumore.

Stanno schiamazzando qua intorno.
Che cavolo è stato?
Cosa è stato questo rumore?
Che spavento!
Mi chiedo che cosa stia succedendo.
Mi è parso di sentire qualcuno qua fuori.
Avete visto quel lampo tutto ad un tratto ?
Cos'era quel rumore?
Forse è caduto qualcosa fuori.
Sembra uno sparo.
Che botta!
Che cos'è stato quel frastuono?
Ma cosa c'è fuori dalla finestra.
Questo rumore mi snerva.
Bisogna andare a controllare fuori, ho sentito qualcosa di strano.
Non si può mai stare tranquilli.

7. Immagina di essere in casa e che ci sia un odore sgradevole:

cos'è questa puzza?
C'è odore sgradevole.
Senti questo odore ?
C'è puzza.
Che puzza!
Che odoraccio!
C'è un cattivo odore in questa stanza.
Che puzza.
Che puzza.
Che puzza.
Che puzza.
C'è un cattivo odore.
C'è un cattivo odore in sala.
Che puzza.
Ma che puzza.
Che fetore!
Che puzza.
Che odoraccio!
In questa casa non si respira dal tanfo.

Che puzza!
C'è qualcosa che puzza.
Cos'è questo odore?
Mi chiedo da dove arrivi quest'odore.
Non c'è un buon odore.
Che odore di fogna qui dentro.
Che è questa puzza?
Soffoco.
Il cavolo è cotto.
Sembra di essere in una fogna.
C'è qualcosa che puzza in cucina.
Non sarà mica andato a male qualcosa.
Ma da dove arriva questo odore.
L?odore qui dentro è insopportabile.

Che cosa è questo odore?
Da dove arriva quest'odore?
Che puzza che c'è qui.
Cos'è che fa questa puzza?
C'è uno strano odore.
Apri la finestra!
Senti anche tu questa puzza?
Sento uno strano odore.
Che tanfo che c'è qui.
Cos'è sto odore?
Questo odore mi da la nausea.
È marcito qualcosa.
Ma cos'è questo tanfo?
C'è aria viziata in camera.

Ma che è morto qualcosa qui dentro?
Ma cos'è questa roba.
Questa casa ha un odore pessimo.
Alfredo hai scorrato?
Sento una strana puzza.
Da dove viene questo cattivo odore?
Da dove proviene questo odore?
C'è un cattivo odore nell'aria.
Che schifo!
Sai come mai ci sia quest'odore?
C'è qualcosa che puzza in questa stanza.

Non sentite questo odore nauseante?
Che sbanfa.
Forse dovrei buttare la spazzatura.
Questo odore è fastidioso.
Che puzza!
C'è odore di gas in cucina.
Che è sto odore?
Non si respira qua dentro.
Credo sia meglio comprare un deodorante per ambienti.
Che tanfo.
Ma che puzza!
L'aria nella stanza è viziata.
Che odore sgradevole!
Sento puzza.

Prendo un deodorante.
Sta bruciando qualcosa?
Mi sembra di sentire puzza di qualcosa.

Che odore insopportabile.
Qualcuno ha mollato?
Sto andando in apnea.

La lettiera del gatto è sporca.
Da dove viene questo odoraccio?
Questo odore è insopportabile.

Che schifo di odore.
Alfred, sei stato tu?
Sarebbe meglio aprire la finestra per fare uscire questa puzza.

Non si può stare qua dentro!

8. Immagina di essere in casa e che ci sia odore di gas:

sento puzza di gas.
Secondo me c'è una perdita di gas.
C'è il gas acceso!
C'è puzza di gas.
C'è il gas aperto?
C'è odore di gas.
Ho paura di aver lasciato il gas aperto.
Non sentite anche voi odore di gas?
Qualcuno ha mollato?
Non respirate!
Anche voi sentire questo odore?
Sento odore di gas.
C'è puzza di gas in cucina.
Che puzza di gas.
Che puzza di gas.
C'è odore di gas.
Avrò lasciato il gas aperto?
Questa puzza è gas?
C'è puzza di metano.
Che puzza di gas!

C'è del gas nell'aria.
Perché c'è odore di gas?
Penso che ci sia odore di gas.
Temo ci sia una fuga di gas.
La cucina perde gas.
Non senti?
Cos'è sto odore?
Rischiamo di saltare in aria.
Se rotto il tubo.
C'è puzza di gas.
C'è un odore che viene dai fornelli.
Spero che la stanza non si sia riempita di gas.
Non respiro.
Che puzza di gas qui!
Che odore strano, da dove verrà?
La senti anche tu la puzza di gas?
Hai chiuso il gas?
Sento cattivo odore.
Abbiamo lasciato il gas aperto?
Sento puzza di gas.
Apri la finestra!
Attenzione, c'è odore di gas.
Ho paura ci sia una perdita di gas.
La stanza è piena di gas.
È odore di gas questo?
Che puzza di gas!
Ci deve essere una perdita.
Avrò lasciato il gas acceso.
C'è un pessimo odore in cucina.
Spero che la caldaia non stia perdendo.
Hai lasciato il fornello acceso, alfred?
Potrebbe esserci una fuga di gas, ne sento l?odore.
Ci sarà una perdita?
Mi sembra ci sia odore di gas.
C'è odore di gas.
C'è odore di gas!
Sento odore di gas.
Chiudi il gas!
Senti anche tu odore di gas?
Sento puzza di gas.
Che puzza di gas.
Ci sono I fornelli accesi?
Controllo un attimo I fornelli.
C'è puzza di gas.
Potrebbe esserci una perdita di gas.
C'è un odore sospetto in casa.
Avrò lasciato I fornelli accesi?
C'è una perdita di gas.
Sento una preoccupante puzza di gas.
Sembra ci sia odore di gas.
Ci deve essere una perdita di gas!

C'è un sentore di gas nella stanza.
Chiudi il gas!
Cos'è questa puzza di gas.
Non accendere I fornelli!
I fornelli sono accesi.
Mi sembra di sentire odore di gas.
Si respira solo gas qui dentro.
Puzza di gas?
Cos'è questo odore sgradevole.
È finita la bombola.
Mi sento soffocare, c'è odore di gas.
C'è puzza in cucina.
Meglio non fumare qui dentro con sta puzza di gas.
Mi serve aria.
Senti. Non trovi ci sia puzza di gas?
C'è qualcosa che non va.

9. Immagina di essere in casa e che ti manchi l'aria:

oddio non riesco a respirare.
Mi manca l'aria.
Mi manca l'aria!
Non riesco a respirare.
Non mi sento bene.
Aiuto, mi sento soffocare!
Mi manca l'aria.
Non respiro.
Non c'è aria qua dentro.
Non respiro.
Non respiro.
Si soffoca.
Non si respira qui dentro.
Soffoco.
Oddio alfred, mi manca l'aria.
Mi manca il respiro.
Non si respira oggi.
Ho bisogno di ossigeno.

Sto morendo asfissiata.
Mi manca il respiro!
Sento che mi manca l'aria.
Mi sento soffocare.
Ho un mancamento.
Non c'è un filo d'aria.
Sto male: ho fame d'aria.
Aiutatemi soffoco!
Aiuto, mi sento soffocare.
C'è troppa polvere, ho l'allergia.
Ho un nodo in gola, mi manca il fiato.
Non si respira in cucina.
Mi sta venendo un attacco di claustrofobia.
Ho bisogno di una boccata d'aria.

Soffoco!
Non mi sento bene.
Sto soffocando.
Non riesco a respirare.
Ho l'affanno!
Non respiro, mi manca l'aria.
Mi manca l'aria.
Mi manca l'aria.
In questa stanza manca l'aria.
Aiuto!sto per soffocare.
Respira affanato.
Sto avendo un attacco di panico.
Non c'è ossigeno nell'aria.
Non si respira.
Mi manca l'aria in questa stanza.
C'è un'aria pesantissima qui dentro.

Mi serve ossigeno.
Aiuto! Mi manca l?aria.
Potrei svenire!
Mi manca l'aria.
Mi sento soffocare.
Non riesco a respirare bene!
Faccio fatica a respirare.
Fatico a respirare.
Non respiro.
Non riesco a respirare.
Faccio fatica a prendere aria.
Soffoco!
Ho bisogno di ossigeno.
Soffoco.
Mi manca l'aria.
C'è puzza in casa!
Mi manca l'aria.
Sto soffocando.
Penso di stare soffocando.

Mi sento mancare.
Aria! Sto male.
Sto boccheggiando.
Mi sento soffocare!
Non respiro bene.
Non respiro.
Fatico a respirare.
Ho il fiato corto.
Mi manca l'ossigeno.
Aiuto!
Questo vestito é troppo stretto.
Ho il naso tappato per il raffreddore.
Non ho aria.
C'è troppo odore di chiuso.
Ho un attacco d'asma.
Aiuto, soffoco.
Mi sento soffocare.
Un po' d'aria non guasterebbe.

10. Immagina di essere in casa e di avere sete:

che sete.
Ho sete.
Che sete!
Ho molta sete.
Ho sete.
Ho molt sete.
Che sete.
Ho sete.
Una zuzza?
Ho la gola secca.
Ho sete.
Ho sete.
Ho la gola secca.
Ho sete.
Che sete.
Ho sete.
Che sete!
Mi è venuta una sete.
Ho la bocca secca.
C'è qualcosa da bere?
Ho bisogno di bere.
Vado a bere.
Ho la gola secca.
Ho la bocca asciutta.
Ho le labbra arse per la sete.
C'ho la gola secca.
Ho bisogno di bere.
Ho bisogno di acqua.
Ho proprio voglia di bere qualcosa.
Ho bisogno di reintegrare un po' d'acqua.
Ci vorrebbe una bibita fresca.
Ho voglia di acqua.
Datemi dell?acqua.
Non ricordo l'ultima volta che ho bevuto.
Mi passi l'acqua.
C'è dell'acqua?
Beviamo qualcosa?
Ho sete e vorrei bere.
Vorrei qualcosa da bere.
Mi serve qualcosa da bere.
Vado a prendere un po' d'acqua.
Ho la bocca asciutta.
Dell'acqua?
Mi scolerei l'intera bottiglia.
Ho la gola secca.
Non bevo da ore.
Ho la bocca impastata.
Mi sento la lingua felpata.
Ho la gola secca.
Ho bisogno di acqua.
Che ho da bere?
Mi prendo un bicchiere d'acqua.
Vorrei qualcosa da bere.
Ho sete!
Ho sete.
Vorrei un bicchiere d'acqua.
Devo bere dell'acqua.
Ho molta sete.
Vorrei un po' di acqua da bere.
Che sete.
Ho la bocca asciutta.
Sono disidratata.
Ho la bocca asciutta.
Mi serve un bicchiere d acqua fresca.
Ho la gola secca.
Mi serve da bere.
Ho la gola secca.
Non bevo da un sacco.
Ho la gola secca.
Non bevo da ieri.
Mi prendo qualcosa da bere.
Ho la gola secca.
Prendo dell'acqua.
Sono disidratata.
Ho la gola secca.
Mi sento la gola secca.
Ho sete.
Sono disidratata.
Non bevo da troppo tempo.
Muoio di sete.
Sto morendo di sete.
Mi piacerebbe un bel bicchiere d'acqua ora.
Muoio di sete.
Che arsura! Questa sete è insopportabile.
Ho la gola secca.

11. Immagina di essere in casa e di dover uscire per andare a fare esercizio fisico:

vado a fare un po' di ginnastica.
Vado a correre.
Vado a fare sport!
Vado a fare una corsetta.
Vado a fare ginnastica.
Devo uscire a correre.
Devo andare a correre.
Torno presto. Vado a fare una corsetta.
Vado in palestra.
Vado un po' a correre.
Odio fare ginnastica quando fa caldo.
Ora mi vesto e vado a correre.
Vado a correre.
Devo uscire a correre.
Devo andare a correre.
Devo andare ad allenarmi.
Dove sono le scarpe?
Preparo la borsa per la palestra.
Esco per fare esercizio fisico.
Esco per andare a fare un po' di sport!
Vado a fare esercizio fisico.
Vado a correre.
Vado a fare esercizio, vuoi venire con me?
Sto per andare a giocare a pallavolo.
Venite anche voi a fare un giro in bici con me?
Io esco che vado a correre.
Sono troppo molla.
Al parco hanno installato un percorso fitness, voglio provarlo.
Vorrei uscire per fare esercizi ma non ho neanche voglia di vestirmi.
Vado a prepararmi per la prova costume.
Avrò preso tutto per la palestra?
Esco a farmi una nuotata alfred.
Vado a fare esercizio fisico.
La parte più difficile è trovare le cose.
Ora vado ad allenarmi.
Oggi ho la lezione in palestra.
Devo andare a fare sport!
Devo andare a correre.
Vado in palestra.
Non voglio stare sempre in casa a dormire, ho bisogno di muovermi.
Devo prepararmi per andare in palestra.
Arrivo dopo un'ora di jogging.
Vado a correre.
Devo dimagrire.
Vado fare una corsa.
Faccio un salto in palestra.
Vado in piscina.
Devo ricordarmi di prendere le scarpe da corsa.
Vado un po' in palestra.
C'è bisogno di un po? Di esercizio.
Non ne ho voglia ma dopo starò meglio.
Esco per andare a correre.
Devo uscire a fare sport.
Esco a fare sport!
Esco e vado a correre.
Esco!
Voglio andare fuori per muovermi un po'.
Tra poco andrò in piscina.
Vieni anche tu in palestra con me?
Vado a nuotare.
Sono fuori forma.
La palestra di luis mi piace tanto.
Vado a fare una corsetta.
Vado in palestra.
Spero di non dimenticare la roba di palestra da nessuna parte.
Alfred, esco a correre.
Inizia una nuova sessione di allenamento.
Non trovo la borsa!
Dove sono le mie scarpe da corsa.
Vado a giocare a tennis.
Vado a correre!
Vado a muovermi un po'.
Vado a fare sport.
Ci vediamo dopo, esco a fare esercizio.
Tra poco ho un torneo di calcetto.
Devo andare in piscina. Mi aspetti quando torno?
Vado ad allenarmi.
Devo sfogarmi facendo un po' di sport.
L'allenamento di oggi sarà intenso.
Oggi mi sento carico, vado a farmi una corsetta.
Vado a migliorare la mia prestanza fisica.
Speriamo non inizi a piovere mentre corro.
Vado a buttare un po' giù la pancia.
Sento bisogno di fare del moto.
Ho tutto quel che mi serve?

12. Immagina di essere in casa e che manchi l'acqua a tavola:

la brocca dell'acqua è vuota.
Non abbiamo messo l'acqua in tavola.
Manca l'acqua!
Non c'è più acqua.
Bisogna prendere l'acqua.
Non c'è l'acqua in tavola!
É finita l'acqua.
C'è poca acqua.
L'acqua?
Che sbadata che sono, manca l'acqua.
Manca l'acqua.
Vado a prendere l'acqua.
È finita l'acqua.
Ho dimenticato la bottiglia d'acqua.
C'è da prendere l'acqua.
Manca l?acqua in tavola.
Manca l'acqua!
Abbiamo finito l'acqua.
Non c'è acqua in tavola.
Non abbiamo messo l'acqua in tavola!
C'è bisogno di altra acqua.
Prendo la bottiglia.
Dov'è l'acqua?
Ho dimenticato l'acqua.
Non abbiamo messo l'acqua in tavola.
La bottiglia d'acqua?
Devo riempire la brocca.
La bottiglia è vuota.
Ho già finito tutta l'acqua in tavola.
Non c'è più acqua in tavola.
Dov'è la bottiglia?
Non c'è da bere.
In tavola non vedo acqua.
È finita l'acqua.
Non c'è più acqua.
L'acqua è ancora in frigo?
Mi passi l'acqua che è in frigo?
Serve dell'acqua in tavola.
Manca l'acqua.
Ci siamo dimenticati l'acqua.
Manca l'acqua.
Ci siamo finiti l'acqua.
Dov'è l'acqua?
Ho dimenticato di comprare l'acqua.
Ho sete.
Con tutto quello che ho mangiato vorrei bere ma non c'è l'acqua.
Manca la brocca dell'acqua.
Ora devo alzarmi per andare a prendere l'acqua.
Alfred, non c'è l'acqua in tavola.
Bisogna portare l?acqua in tavola.
Servirebbe qualcosa per mandare giù questo boccone.
Prendi l'acqua che è finita?
Manca l'acqua.
Puoi prendere l'acqua?
È finita l'acqua.
Non c'è l'acqua.
Serve dell'acqua.
Non c'è più acqua.
La bottiglia dell'acqua è vuota.
Qualcuno può prendere l'acqua?
Qualcuno deve tornare in cucina.
L'acqua nel frigo.
Ho sete, ma non c'è l'acqua in tavola.
Non c'è da bere a tavola.
Dopo la pasta ho sempre bisogno di un bicchiere d'acqua.
Manca l'acqua.
Non c'è acqua in tavola.
Ho sete.
Bisogna andare a prendere da bere.
Abbiamo apparecchiato senza ricordarci l'acqua.
Non c'è l'acqua!
Ho finito tutta l'acqua.
Prendo l'acqua.
Vorrei dell'acqua fresca.
Ho lasciato l'acqua in frigo.
Ti sei dimenticato di portare l'acqua.
Una brocca d'acqua?
Vedo che eravate assetati.
È finita l'acqua.
Dove ho messo la bottiglia d'acqua.
Mancano le bottiglie in tavola.
Ho preso tutto meno la bottiglia d'acqua quando ho messo tavola.
Ho il bicchiere vuoto.
Come mai non c'è acqua?
La caraffa è vuota.

13. Immagina di essere in casa e che si sia sviluppato un piccolo fuoco libero:

al fuoco!
Al fuoco!
Spegni!
A fuoco.
Va a fuoco!
Aiuto, c'è un fuoco!
La presina ha preso fuoco!
Aiuto ci sono delle fiamme.
Aiuto!
Va a fuoco!
Sento puzza di bruciato.
Al fuoco!
Ci sono I fornelli che vanno a fuoco.
Oddio un incendio.
Oddio un incendio.
C'è del fuoco.
Aiuto c'è il fuoco!
Va a fuoco!
Ho incendiato il tappeto.
Aspetta, lo spengo!
Allarme incendio.
Serve un secchio!
Si è acceso un fuoco!
Qualcosa ha preso fuoco!
A cosa avete dato fuoco ?
Presto, qualcuno mi aiuti.
Attenti che vi bruciate.
Sta bruciando la presa.
Brucia tutto!
L'allarme antincendio è scattato in salotto!
Dov'è un estintore quando serve.
Sta bruciando la casa.
Si sta sviluppando un piccolo fuoco!
La situazione sta diventando scottante.
Presto sta prendendo fuoco.
Qualcosa ha preso fuoco!
Si è acceso un piccolo fuoco!
Sta bruciando qualcosa.
Una coperta!
Bisogna spegnere il fuoco!
Esce del fumo dal forno!
Aiuto. Sta bruciando qualcosa.
Fuoco!
Allontanatevi da lì.
Aiuto un fuoco.
Passami quell'acqua!
Il mio ufficio è in fiamme!
Non ricordo nessun metodo per spegnere I fuochi.
Aiuto, le fiamme.
Quello è un pericoloso fuoco!
Delle fiamme !?
Oddio ha preso fuoco.
C'è un incendio in cucina.
Oddio!
Qualcosa brucia.
Serve dell'acqua!
Presto, spegniamo il fuoco prima che si propaghi!
Esce del fumo dalla cucina!
C'è un principio di incendio.
Prendete un estintore.
Sta bruciando tutto!
Non voglio ustionarmi.
Ha preso fuoco qualcosa!
C'è la cucina che va a fuoco.
Cavolo non ho niente a portata di mano per spegnere il fuoco.
Al fuoco, al fuoco!
Attenzione! Fuoco!
Quale è il numero dei pompieri?
Si è incendiato lo strofinaccio.
La cucina sta bruciando.
Spegniamolo!
C'è un piccolo incendio.
L'estintore!
Butta qualcosa sopra il fuoco!
Va a fuoco qualcosa!
Vedo delle piccole lingue di fuoco.
Portate dell'acqua.
C'è stato un cortocircuito !
Sta suonando l'allarme antincendio.
Come ha fatto a prendere fuoco così?!
C'è un fuoco in dispensa!
Oh cavolo mi va a fuoco la casa.
Si sta incenerendo tutto.
Al fuoco!
Al fuoco!

14. Immagina di essere in casa e che si sia stato un guasto alle tubature tale per cui non c'è più acqua corrente:

Non c'è più acqua.
Le tubature sono guaste.
Non c'è più l'acqua!
Non esce più acqua dal rubinetto.
Non c'è acqua.
Non scende l'acqua.
É andata via l'acqua.
Non viene più acqua dal rubinetto.
Mi spiace, ma non c'è acqua.
Non c'è più acqua,mi dispiace.
Non posso lavarmi.
Manca l'acqua, potrebbero essersi rotti I tubi.
Non esce acqua dai rubinetti! Ci sarà un guasto!
Cosa è successo all'acqua?
Manca l'acqua in casa alfred.
Non c'è acqua.
Non è possibile non avere acqua.
Non arriva acqua.
Non esce acqua dal lavandino.
Siamo senz'acqua!

Non c'è più acqua.
Non va l'acqua.
Cos'è successo ai tubi?
Siamo rimasti senz'acqua corrente.
Non abbiamo l'acqua per lavarci.
Si son rotte le tubature.
Nel deserto ci sarà più acqua che qui.
Senza acqua è un disastro.
Non c'è acqua, chissà se manca anche ai vicini.
La doccia non funziona più! Sarà per un guasto ai tubi.
Spero non sia un problema dell'acquedotto.
Siamo senza acqua.
L?acqua non viene più.
C'è un guasto?
L'acqua è staccata.
Hanno chiuso l'acqua nella via per un guasto.
Uffa dev'esserci stato un guasto alle tubature.

Non scorre acqua dal rubinetto.
C'è un problema ai tubi.
Non posso lavarmi le mani, siamo senza acqua.
Fino a domani saremo senz'acqua.
Hanno chiuso l'acqua.
Le tubature sono rotte, siamo a secco.
Stanno facendo dei lavori per strada.
C'è un guasto alle tubature.
Quando apro il rubinetto sento uno strano gorgoglio e l'acqua non scende.
Il bagno è allagato, dev'essersi rotto un tubo.
Ora come faccio a lavarmi senz'acqua?
Non c'è acqua per lavarsi.
Siamo rimasti senza acqua corrente.
Non c'è acqua!
Non esce acqua dal rubinetto.
Non c'è acqua corrente in casa.

Uffa, è andata via l'acqua!
Si sono rotte le tubature.
Non esce acqua dai rubinetti.
Ci serve un idraulico, si sono rotti I tubi.
Manca l'acqua.
Manca l'acqua.
Siamo senz'acqua.
Dobbiamo comprare le bottiglie d'acqua.
Non c'è acqua.
Bisogna chiamare l'idraulico.
Siamo rimasti senz'acqua a causa di un guasto.
Sapevo che dovevo far controllare I tubi.
Che succede all'acqua?
Penso sia saltato un tubo. Non scorre acqua.
Chissà quanto durerà?
C'è un problema con l'acqua.
Vorrei fare la doccia ma non c'è più acqua in casa.

Non abbiamo più acqua corrente!
L'acqua non riesce ad uscire dal rubinetto.
Hanno chiuso l'acqua?
Mannaggia, siamo rimasti senz'acqua.
Non esce più acqua dal lavandino.
Non possiamo fare la doccia. C'è un guasto all'impianto.
Il bagno non funziona.
Devo chiamare l'idraulico!
Non esce acqua dal rubinetto.
Volevo farmi una doccia ma a quanto pare manca l'acqua.
In giardino non c'è più acqua! S'è rotto un tubo.
Dovrei chiamare un idraulico.
Non viene l'acqua.
I rubinetti sono a secco. Si sarà rotto un tubo.
Staranno facendo dei lavori?

15. Immagina di essere in casa e di doverti metterti a cucinare:

Cosa preparo per pranzo?
Vado a preparare la cena.
Cuciniamo?
Adesso faccio qualcosa da mangiare.
Cucino qualcosa.
Vado in cucina a preparare qualcosa.
Vado a cucinare.
È ora di preparare la cena.
Cucino.
È quasi ora di cena, vado a cucinare.
Preferite penne o spaghetti?
Ho fame ma non ho voglia di cucinare.
Devo preparare da mangiare.
Ho fame.
Cominciamo a cucinare.
Devo cucinare.
Come era quella ricetta?
Sarà pronto tra mezz'ora più o meno.

Ho fame, vado a preparare.
Cosa mangiamo?
Dobbiamo mangiare, cucino la cena.
Preparo il pranzo.
Ho fame, cucino qualcosa.
Inizio a preparare qualcosa.
Cosa preferite per cena? Così preparo.
Io vado in cucina.
Aiutatemi, non voglio avvelenarvi.
Spero di salare correttamente la pasta.
Devo iniziare a preparare qualcosa per la cena di stasera.
Mi devo metterò sotto con la cucina.
Avrò abbastanza ingredienti in casa per cucinare?
Mi metto ai fornelli.
Devo cucinare qualche ricetta prelibata.
Ho finito il sale!

Va bene se faccio una pasta?
Cosa faccio per cena?
Cosa faccio da mangiare?
Preparo la cena.
Preparo la cena.
Vuoi qualcosa?
Preparo del cibo.
Vado a mettere su qualcosa.
Mi devo mettere a preparare qualcosa da mangiare.
Preparo qualcosa.
Adesso mi metto il grembiule.
Sono una cuoca provetta.
Cuciniamo qualcosa.
Mi devo mettere ai fornelli.
Sono impedita a cucinare.
Vado in cucina.
Devo mettermi ai fornelli.
Dove è quella pentola?
Dov'è la pasta?

È ora di pranzo, vado a cucinare.
Cucino qualcosa!
Mi metto a cucinare.
Faccio da mangiare.
Ho fame, mi faccio qualcosa.
Vado a preparare qualcosa da mangiare.
Mi fai compagnia mentre cucino?
Faccio da mangiare.
Vi preparo un bel pranzo.
Stasera cucino io.
C'è qualcosa in frigo?
Dovrei cucinare un po' di carne.
Non ho voglia di cucinare.
Mi servono alcuni ingredienti per cucinare.
Spero ci sia qualcosa da cucinare.
Mi manca un ingrediente.
Non ho voglia di cucinare.
Ho voglia di cucinare.

Mi metto a cucinare!
Devo preparare qualcosa da mangiare.
Devo cucinare.
Cosa vuoi che cucini?
Inizio a mettere su l'acqua per la pasta.
Cosa mangiamo stasera? Così mi metto a farlo.
Vado a cucinare.
Mi metto all'opera !
Spero vi piaccia la mia cucina.
Cucinerei ma in casa non c'è niente da mangiare.
Devo accendere I fornelli e cucinare qualcosa di buono.
Cucino qualcosa di fresco o di caldo?
Preparo qualcosa, alfred.
Vado a cucinarmi qualcosa.
Devo andare a fare la spesa.

16. Immagina di essere in casa e di doverti informare sugli eventi di attualità:

Notizie dal mondo?
Cosa sta succedendo nel mondo?
Cosa succede nel mondo?
Devo leggere il giornale.
Guardiamo il tg.
Vado a comprare il giornale.
Sapete se oggi è successo qualcosa?
Cosa è successo oggi nel mondo?
Che si dice di nuovo?
Non guardo il telegiornale da tanto, devo aggiornarmi.
A che ora è il tg.
Cosa succede nel mondo?
Devo guardare il tg.
Chissà che news ci sono oggi.
Vediamo che succede nel mondo.
Devo aggiornarmi sull'attualità.
Chissà cosa succede oggi?
Hai sentito le notizie di oggi?
Non ho idea di cosa sia accaduto in italia oggi.
Che notizie ci sono?
Accendo la televisione per sapere le notizie.
Cerco su internet.
Prendo il giornale, così sarò più informata.
Avete letto il giornale oggi?
Voglio informarmi sui fatti di questa giornata.
Sono un cavernicolo, non so cosa sta succedendo là fuori.
Il giornale oggi è in offerta.
Hai letto I giornali oggi?
Dovrei controllare il telegiornale di oggi.
Controlliamo un po' cosa dice la bbc.
Vediamo il giornale cosa dice.
Leggo sul giornale per sapere gli ultimi eventi.
Hai sentito le ultime?
Hai comprato il giornale?
Devo sapere cosa sta capitando attualmente.
Vado a vedere cosa è successo oggi nel mondo!
Mamma metti il telegiornale.
Mi servirebbe sapere questa cosa.
Sono proprio fuori dal mondo, mi devo informare sull'attualità.
È successo qualcosa oggi?
Cosa hanno al telegiornale di stamattina?
Cerco qualche evento in città a cui possiamo partecipare.
Non ho ancora sentito le notizie di oggi.
Voglio sempre essere informato su cosa succede intorno a me.
Dovrei vedere sul telefono se sia successo qualcosa di importante oggi.
Non ho ancora letto le news oggi.
Vediamo cosa si dice sul web.
Chissà cosa sta succedendo.
Non ho ancora visto il tg oggi.
Hai visto il telegiornale?
Devo informarmi sugli eventi di attualità.
Cosa dicono I giornali?
Cerco delle notizie su internet.
Prendo il giornale.
Voglio essere informata sull'attualità.
Avete visto il telegiornale oggi?
Avete notizie di oggi?
Chissà come è finita la partita di calcio?
L'applicazione di google edicola è utile.
Qualche nuova notizia?
Dovrei cercare gli eventi di oggi sul web.
Quasi quasi guardo il telegiornale.
Alfred, novità interessanti?
Accendo il tg: voglio sapere cosa succede.
Che accadrà oggi?
È successo qualcosa di interessante ?
Voglio scoprire le notizie di attualità.
Di cosa parlano I telegiornali?
Ascolto la radio per conoscere le notizie.
Leggo il giornale.
Mi informo su internet.
Cosa è successo oggi?
Che novità ci sono ?
Dovrei comprare dei giornali.
Sto guardo il tg.
Ora mi cerco due notizie.
Dovrei leggere il giornale di oggi.
Che casini saranno successi nel mondo oggi.
Sintonizziamoci sul telegiornale.
Dovrei sapere di più sugli eventi di attualità.
Niente di nuovo?

17. Immagina di essere in casa e di voler guardare un film o una serie tv:

Che film guardiamo?
Voglio vedere un film.
Mi guardo un film!
Ho voglia di vedere un film.
Voglio vedere la tv.
Ho proprio voglia di vedere questa serie.
Guardiamo un film?
Non vedo l'ora che cominci.
Guardiamo un film.
Devo recuperare qualche episodio di quella serie.
Per fortuna ho netflix.
Voglio vedere un film.
Vorrei finire quel film che avevo iniziato.
Quasi quasi accendo la tivù.
Vediamo cosa c'è di bello in tv.
Vorrei guardare un film.
Ci vorrebbe un bel film.
Ci guardiamo un film insieme?
Stasera esce il film del mio attore preferito.
Ti va di vedere un film?
Che film vuoi vedere?
Apro netflix.
Non vedo l'ora di vedere che succede in questa serie.
Avete una serie tv da consigliarmi?
Accendiamo la tv: cominciano I nuovi episodi.
Filmetto?
È uscita la nuova puntata di questa serie.
I film gialli sono I miei preferiti.
Ho proprio voglia di rilassarmi con un bel film.
Oggi c'è la mia serie preferita in tv.
Vediamo cosa c'è di nuovo su netflix.
Che noia, mettiamo sul canale dei film.
Vediamo che film c'è in programmazione.
Chissà che c'è oggi in tv.
Cosa danno in tv questa sera?
C'è una bella serie tv in televisione.
Voglio vedere un film!
Mettiamo una nuova serie ?
Accendo la tv.
Per staccare un po' mi serve un film.
Avete un film da consigliarmi?
È uno dei film migliori. Non bisogna perderlo in tv.
Film?
Devo recensire questo film, guardiamolo.
Oggi maratona film.
Voglio iniziare una nuova serie tv.
Questa serie dev'essere bellissima.
Finalmente posso vedere l'ultimo film del mio regista preferito.
Alfred, ci guardiamo un film?
Guardiamo una serie tv o un film.
Potrei guardare una serie tv.
Ti va di guardare stranger things?
Mi annoio, devo scegliere un film.
Ci guardiamo un film?
Mi piacerebbe vedere un film.
Vorrei vedere un film.
Ora mi guardo un film.
Conoscete qualche film carino da vedere?
Ti piace questa serie? La guardiamo?
Tv?
Non troveremo mai un film che non abbiamo già visto tutti.
Adoro le serie tv.
In tv non c'è niente, guarderò netflix.
Vorrei vedere questo film che è uscito da poco.
Mi son persa

l'ultimo episodio della mia serie preferita.
Accendiamo il computer e guardiamo una serie.
Potrei guardare una serie tv.
Ci starebbe un episodio.
Cosa potremmo guardare?
Guardiamo qualcosa in televisione.
Ti va se ci guardiamo un film?
Cosa hai voglia di vedere alla tivù?
Guardo qualcosa.
Ho bisogno di vedere un film.
Mi consigliate un film?
In tv stanno per trasmettere una prima visione del film che ho perso al cinema.
Netflix?
Questo film è molto acclamato dalla critica.
Il signore degli anelli quando inizia?
Ora mi guardo la nuova serie di sky.
Sta sera si guarda il mio film preferito.
Controllo l'indice programmi così vedo se c'è qualcosa che mi interessa.
Sul canale delle serie ci sarà qualcosa di interessante?
Quale serie tv staranno trasmettendo?
Speriamo ci sia qualcosa di interessante.

18. Immagina di essere in casa e di aver bisogno di rilassarti o di svagarti:

Cosa potrei fare per svagarmi?
Necessito di molto relax.
Ho bisogno di riposarmi!
Non voglio pensare a niente oggi.
Mi serve un po' di relax!
Andiamo a fare qualcosa, voglio svagarmi.
Ho bisogno di un periodo di riposo.
Accendi lo stereo. A questo punto ci vuole un po' di buona musica per distrarsi.
Che stanchezza.
Potremmo andare a correre per svagarci.
Facciamo un break.
Cosa potrei fare per passare il tempo?
Vorrei prendere un po' di sole in spiaggia.
Ora mi piazzo sul divano e cazzeggio.
Voglio spassarmela.
Devo staccare un po?: Spero di rilassarmi.
Ho voglia di fare qualcosa di divertente.
Ho bisogno di rilassarmi un po'.
Cosa posso fare per svagarmi?
Mi riposo un momento!
Vorrei tanto rilassarmi tutto il giorno.
Stacco un attimo.
Ora mi rilasso.
Ho bisogno di una vacanza.
Mi piacerebbe fare una partita a carte.
Ho bisogno di staccare un po'.
Sono stanca, oggi non voglio alzare un dito.
Quasi quasi mi faccio un bagno.
Non ne posso più, sono troppo stressato, devo rilassarmi.
Potrei fare un pisolino.
Potrei giocare un po' ai videogiochi.
Ho bisogno di svagarmi.
Vorrei svagarmi.
Sono proprio stanco.
Facciamo qualcosa di rilassante?
Ho davvero necessità di fare una pausa.
Voglio prendermi una pausa!
Ho bisogno di una pausa relax.
Mi serve una pausa.
Sono affaticata, devo rilassarmi.
Ci vorrebbe un bel massaggio rilassante.
Se mi passi il cruciverba, ne facciamo uno insieme.
Giochiamo a qualcosa?
È una bella giornata per un po' di mare e relax.
Sono stanca.
Ahhh, finalmente un po'di relax!
Potrei ascoltare un po' di musica.
Mi va di suonare un po' non ho più voglia di lavorare.
Adesso mi rilasso un po'.
Ho bisogno di un po? Di relax.
Sono a pezzi.
Mi riposo un po'.
Devo riuscire a pensare ad altro.
Mi prendo un attimo per svagarmi!
Ora mi stendo e mi rilasso.
Voglio riposarmi.
Voglio svagarmi un po'.
Ho bisogno di una pausa.
Vorrei leggere il mio best seller preferito.
Ho bisogno di sbattere.
Ho bisogno di leggere un libro per rilassarmi.
Ho voglia di un massaggio.
Sono stanchissimo, mi rilasso un po' e poi me ne vado a dormire.
Vorrei leggere un bel libro.
Vediamo se trovo qualche video divertente.
Ho voglia di divertirmi.
Se non stacco impazzisco.
Non ce la faccio più.
Vado a camminare per svagarmi.
Ho bisogno di rilassarmi.
Voglio rilassarmi un po'!
Ho bisogno di rilassarmi un po'.
Voglio rilassarmi.
Devo staccare la spina, rilassarmi un po'.
Facciamo una passeggiata?
Sento il bisogno di un po' di relax.
Che stress.
Mi farei un sonnellino volentieri.
Ho proprio bisogno di fare una pausa.
Ho lavorato tutto il giorno, ora mi voglio rilassare.
Dovrei guardare un bel film tranquillo.
Ora mi riposo un po'.
Alfred, giochiamo?
Devo rilassarmi.
Ho bisogno di staccare.

19. Immagina di essere in casa e di essere annoiato:

Tu non ti stai annoiando?
Che palle, mi annoio.
Che noia mortale!
Sono così annoiata.
Uffa!
Facciamo qualcosa, muoio dalla noia.
Sto morendo di noia.
Che pizza questo tempo uggioso.
Sembra domenica.
Che barba.
Che barba, che noia!
Non trovo un modo per passare il tempo.
Adesso prendo qualcosa per divertirmi.
Mi viene da sbadigliare dalla noia.
Non so cosa fare: che noia.
Che giornata noiosa!
Non so cosa fare.
Che noia in questa casa.
Uffa, che palle!
Sono molto annoiata, non so cosa fare.
Non ne posso più.
Mi sto annoiando molto.
Facciamo qualcosa?
Che barba stare in casa tutto il giorno.
Che palle.
Che barba !
Facciamo qualcosa.
Queste giornate sono tutte uguali.
Non so che fare oggi.
Devo fare qualcosa.
Mi sto annoiando.
Mi annoio.
Che barba!
Mi sto rompendo qui.
Non c'è nulla da fare in casa.
Mi sto annoiando a morte!
Mi sono rotta di stare qui a non far niente.
Non so che fare.
Non ho niente da fare.
Dobbiamo trovare qualcosa da fare.
Mi sto annoiando.
Qualcosa da fare?
Questo tempo mi fa passare la voglia di fare cose.
Questo film è noiosissimo.

Dovrei trovarmi qualcosa da fare, mi annoio.
È da un ora che guardo il muro.
Vediamo se c'è qualcosa di interessante da fare.
Mi sento solo.
La noia mi sta ammazzando.
Non so che fare!
Non c'è nulla di divertente.
La vita è troppo monotona.
Non so che fare!
Non ho voglia di fare niente.
Che rottura.
Non so cosa fare, mi annoio.
Non c'è nulla da fare.
Non mi sto divertendo per nulla.
Voi?
Uff.
Non so cosa fare.
Odio stare a casa quando piove.
È una noia mortale.
Sto morendo di noia.
Ci sarà qualcosa da fare in casa.
Sto morendo di noia.
Che barba che noia, che noia che barba.
Che giornata monotona!
Che noia.
Mi annoio.
Che noia!
Non c'è la faccio più a stare in casa.
Che noia!
Che noia oggi.
Che noia.
Non so cosa fare.
Avete idee?
Che noia.
Mi annoio.
Che noia.
Che noia, non so che fare.
Non c'è niente da fare.
Mi annoio.
Che noia.
Che noia.
Che noia!

20. Immagina di essere in casa e di non riuscire a dormire:

Sono troppo emozionato per dormire.
Ho bisogno di dormire ma non ci riesco.
Non ho sonno!
Non mi trovo.
Sono agitato.
Che fastidio, non riesco ad addormentarmi.
C'è troppo rumore per dormire.
Mi giro e rigiro nel letto senza dormire.
Ho troppi pensieri per la testa.
Sono agitata.
Fa troppo caldo, non riesco a prendere sonno.
C'è troppo caldo, non riesco a dormire.
Se metto del rumore bianco mi viene sonno sicuro.
Mi giro e rigiro nel letto ma non mi addormento.
Vorrei addormentarmi.
C'è della camomilla?
Non riesco a prendere sonno.
Come faccio a prendere sonno?
Non riesco a prendere sonno!
Non riesco ad addormentarmi.
Non ho sonno.
Ho l'insonnia.
Non riesco a prendere sonno.
Non ho chiuso occhio dalle due.
Non riesco a chiudere occhio.
Con questo caldo non riesco a dormire.
Non riesco a prendere sonno.
Ho bevuto troppo caffè oggi.
Dovrei provare a rilassarmi.
Non riesco a prendere sonno.
Sono troppo sveglio!
Vorrei dormire.
Eppure oggi ero così stanco!
Ho troppi pensieri per la testa.
Non mi sto addormentando.
Non riesco ad addormentarmi!
Soffro d'insonnia.
Non riesco a prendere sonno.
Vorrei dormire, ma non riesco.
Fa troppo caldo per dormire.
Non riesco a prender sonno.
Sono scomoda.
C'è troppo rumore.
Non ne posso più, sono ore che mi rigiro nel letto.
Dovevo svegliarmi prima.. Ora non ho sonno.
Magari se ascolto un audiolibro mi addormento.
Sono troppo sveglio.
Non riesco a dormire.
Che scomodo questo letto.
Non riesco a dormire.
Non riesco a dormire!
Non riesco a chiudere occhio.
Non riesco a dormire.
Non riesco a dormire.
Non riesco a dormire.
Tu dormi ?
Uffa non riesco a dormire.
Soffro di insonnia.
Quasi quasi mi faccio una camomilla.
Non riesco a dormire.
Non ho sonno.
Non riesco a dormire.
Stanotte non riesco a dormire.
Non riesco a prendere sonno.
Si prospetta una notte in bianco!
In questa casa ci sono troppi rumori.
Sono assonnata ma non riesco ad addormentarmi.
Non ho voglia di dormire!
Non riesco ad addormentarmi.
Non riesco a rilassarmi.
Non so come mai, ma non dormo.
C'è troppa luce per dormire.
È da un'ora che sono in letto e sono ancora sveglio.
Sono in ansia per domani.
Sto subendo ancora l'effetto del jet lag.
Sono così stanco che non riesco a dormire!
C'è una zanzara che non mi fa dormire.
Mi passasse sta insonnia.
Alfred, sono insonne.
L?insonnia mi tormenta.
Notte insonne!

21. Immagina di essere in casa e che ci sia troppo silenzio:

Non mi piace tutto questo silenzio.
La quiete in questa stanza è eccessiva.
Non si sente un rumore!
Che cosa state facendo? Dove siete?
C'è qualcuno in casa?
Che tranquillità!
Mettiamo un po' di musica in sottofondo.
Dimmi qualcosa.
Questo silenzio mi da fastidio alle orecchie.
È fastidioso questo silenzio.
Non amate fare conversazione?
Con queste nuove finestre c'è un silenzio assoluto.
Non si sente nemmeno un battito d'ali.
Mi sento solo mi piacerebbe ci fosse un po' di rumore.
Non vola una mosca.
C'è troppo silenzio.
Che silenzio!
C'è un silenzio imbarazzante.
Non vola una mosca in questa stanza.
Che pace che c'è!
Non si sente volare una mosca!
Come mai c'è così tanto silenzio?
Questo silenzio mi fa angosciare.
C'è troppo silenzio.
Non si sente volare una mosca.
Rompiamo il ghiaccio.
Non si sente volare una mosca.
C'è troppo silenzio, dovrei mettere un po' di musica.
Non vola una mosca.
C'è troppo silenzio.
Nessun rumore.
C'è un silenzio assordante.
Sembra un cimitero!
Mi fischiano le orecchie dal silenzio.
C'è troppa calma in questa casa.
Che tranquillità!
Perché non mi parla nessuno?
Ehi!

Non si muove una foglia.
Non vola una mosca.
C'è un silenzio tombale.
Un po' di musica di sottofondo non sarebbe male.
C'è un silenzio di tomba.
Che silenzio.
Non c'è una bava di vento.
Che depressione sta casa.
Non c'è nessuno.
Questo silenzio è assordante.
C'è troppo silenzio.
Non c'è troppo silenzio?
Non mi piace questa totale assenza di rumore.
Non vola una mosca!
È tutto così silenzioso.
C'è qualcuno sveglio?
Finalmente un bel silenzio.
Che silenzio.
Il silenzio fa quasi rumore.
Riesco a sentire gocciolare il rubinetto.
C'è troppo silenzio.
Non vola una mosca.
Sembra che non ci sia anima viva qui.
Mi piacerebbe sentire un po' di casino.
Che silenzio.
Silenzio di tomba.
Tutto tace?
Che silenzio.
C'è troppo silenzio.
Che silenzio!
Ehi c'è qualcuno?
Che silenzio!
Che silenzio!
C'è un silenzio inquietante.
È nato un frate.
Che silenzio imbarazzante.
I silenzi mi imbarazzano.
C'è un silenzio di tomba.
Non si sente un suono.
Mettiamo un po' di musica.
Questo silenzio è inquietante.
Che silenzio.
Non vola una mosca!

Appendice C

DialogFlow SDK

Il *Software Development Kit* (SDK) fornito da DialogFlow permette di integrare in nell'interfaccia desiderata le potenzialità del proprio *agent* mediante l'implementazione di funzioni in un linguaggio di programmazione compatibile (nel progetto si è utilizzato il linguaggio Python). L'SDK permette di sviluppare applicazioni con tre usi tipici: *fullfillment*, *detect intent API* ed *agent API*.

Il *fulfillment* è utilizzato nei sistemi a base conversazionale e serve a collegare l'*agent* Dialogflow ai servizi, API e/o database. È utile per connettere le funzionalità di Dialogflow legate all'elaborazione del linguaggio naturale con altri *backend*, API e database al fine di creare interfacce conversazionali contestuali, complesse e personalizzate [13]. È possibile utilizzare il *fulfillment* ad esempio per ordinare oggetti o per recuperare informazioni legate all'utente quali quelle contenute in e-mail ed altre risorse testuali.

Le *detect intent API* permettono di incorporare le funzionalita dell'*agent* DailogFlow in software, app o siti web che necessitano di una interfaccia conversazionale. Questo tipo di API consentono quindi di eseguire interrogazioni all'*agent* mediante richieste audio o testuali da parte dell'utente, ricevendo da esso una risposta.

Le *agent API* infine, permettono di gestire dinamicamente il comportamento dell'*agent* creando, leggendo, aggiornando ed eliminando gli *intent*, le *entity* ed i *context* in esso (vedi dialogflow.com/docs/reference).

Nella Figura C.1 è illustrato lo schema generale di funzionamento di un *agent* DialogFlow con evidenziate le tre funzionalità descritte qui sopra. Il *fullfillment* può essere implementato in qualsiasi piattaforma che supporti la gestione delle *HTTP requests*. Sia le *detect intent API* che le *agent API*, invece, sono accessibili tramite l'`API REST` di Dialogflow o le librerie client (`V2`, `V2beta1` e `V1`).

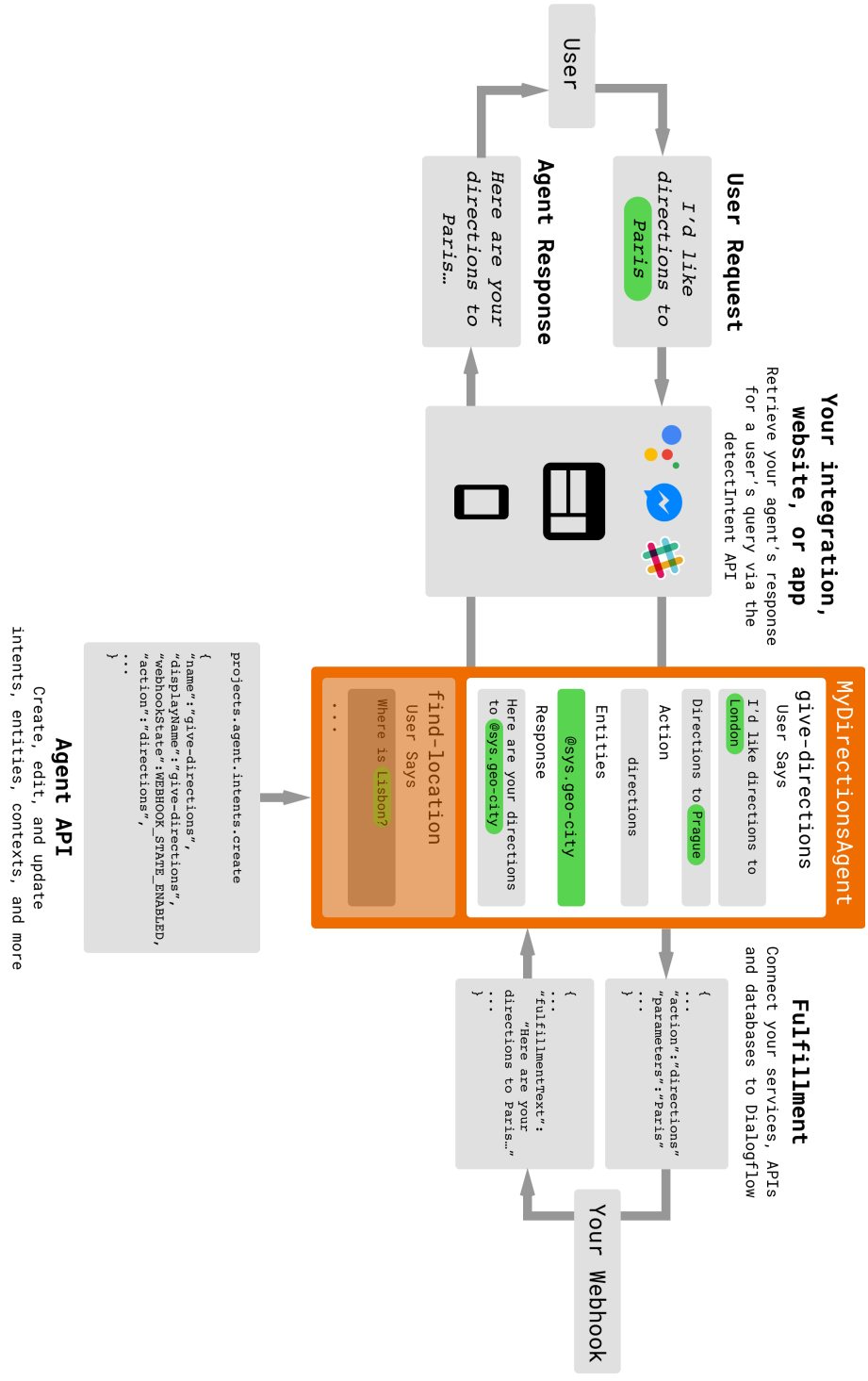


Figura C.1: Schema generale di funzionamento di un *agent* in DialogFlow (enfasi su API ed SDK) (immagine da dialogflow.com/docs/sdks).

Appendice D

File JSON

D.1 Impostazioni

Il file `settings.json` contiene tutte le informazioni necessarie per il collegamento dei *software* che sfruttano gli SDK di DialogFlow alla piattaforma stessa. Viene qui riportato, per completezza, un *template* generale (utilizzato per generare i file `settings.json` dedicati ai singoli *agent* mediante lo script `settingsGenerator.py`, come visto in Figura 4.2) in cui i valori `AGENT_NAME`, `CLIENT_ACCESS_TOKEN` e `DEVELOPER_ACCESS_TOKEN` sono da adattare all'*agent* desiderato.

```
{
    "starting-language": "italian",
    "dialogflow": {
        "agent": "AGENT_NAME",
        "default-language": "it",
        "url": "https://api.dialogflow.com/v1/intents",
        "protocol-version": "?v=20150910",
        "intents-folder": "Dialogflow-Intents",
        "tokens": {
            "AGENT_NAME": {
                "client": "CLIENT_ACCESS_TOKEN",
                "developer": "DEVELOPER_ACCESS_TOKEN"
                }
            }
    },
    "followups": {
        "it": {
            "yes": [
                "fallo",
```

```
            "esatto",
            "certo",
            "buona idea",
            "confermo",
            "d'accordo",
            "certamente",
            "proprio così",
            "sicuro",
            "sì",
            "assolutamente sì",
            "sì, grazie",
            "sì, per favore"
        ],
        "no": [
            "non mi interessa",
            "non proprio",
            "non credo",
            "non voglio",
            "non farlo",
            "non sono d'accordo",
            "non penso",
            "certo che no",
            "no",
            "assolutamente no",
            "no, grazie"
        ]
    },
    "en": {
        "yes": [
            "yes",
            "yes, please",
            "yes, thanks",
            "do it",
            "sure",
            "exactly",
            "confirm",
            "of course",
            "sounds good",
            "that's correct",
            "I don't mind",
            "I agree",
            "ok"
        ],
        "no": [
            "no",
            "no, thanks",
            "don't do it",
            "definitely not",
```

```
                "not really",
                "thanks but no",
                "not interested",
                "I don't think so",
                "I disagree",
                "I don't want that"
            ]
        }
    }
}
```

D.2 Output JSON

DialogFlow restituisce come risultato della interrogazione da parte dell'utente all' *agent* sia l'*intent* identificato, visibile anche dalla web console del sito internet, sia una risposta "completa", in formato `json`. Come si può notare nel risultato sono presenti i dati sia per quanto riguarda l'`intentName` identificato, sia per quanto riguarda le altre funzionalità implementabili mediante la piattaforma, quali `action`, `parameters` e `contexts`.

Per primo viene riportato come esempio il risultato di una interrogazione ad un *agent* predefinito, l'*agent* `weather`, in relazione a quanto visto nella Sezione 3.1.1.


```
{
  "id": "e3e57311-3d7a-4dbe-8381-e0d9cabbc33f",
  "timestamp": "2018-07-19T14:18:10.329Z",
  "lang": "it",
  "result": {
    "source": "agent",
    "resolvedQuery": "secondo te posso correre domani a londra?",
    "action": "weather.activityAction",
    "actionIncomplete": false,
    "parameters": {
      "activity": "run",
      "date-time": "2018-07-20",
      "address": {
        "city": "Londra"
      }
    },
    "contexts": [
      {
        "name": "weather",
        "parameters": {
          "date-time.original": "domani",
          "activity.original": "correre",
          "address": {
            "city": "Londra",
            "city.original": "londra?"
          },
          "activity": "run",
          "date-time": "2018-07-20",
          "address.original": "londra?"
        },
        "lifespan": 2
      }
    ],
```

```
    "metadata": {
      "intentId": "26ed0882-8fc2-4821-9bc8-fd6a182e7aca",
      "webhookUsed": "false",
      "webhookForSlotFillingUsed": "false",
      "isFallbackIntent": "false",
      "intentName": "weather.activity"
    },
    "fulfillment": {
      "speech": "",
      "messages": [
        {
          "type": 0,
          "speech": ""
        }
      ]
    },
    "score": 0.8299999833106995
  },
  "status": {
    "code": 200,
    "errorType": "success"
  },
  "sessionId": "70d93a5e-b73d-c779-0ad1-ce0da67f251d"
}
```


Come secondo esempio viene riportato il file restituito dal *client* Windows sviluppato, visto nella Sezione 3.4, il cui codice (`client.py`, vedi Figura 3.3) è scritto in modo da restituire il seguente file `answer.json`.


```
{
    "id": "9a78f380-d8bd-4a54-9d0d-bae3ec6f5102",
    "timestamp": "2018-07-19T14:43:31.462Z",
    "lang": "it",
    "result": {
        "source": "agent",
        "resolvedQuery": "c'\u00e8 davvero moltissima afa oggi",
        "action": "",
        "actionIncomplete": false,
        "parameters": {},
        "contexts": [],
        "metadata": {
            "intentId": "763b8abc-95f5-4ff0-a8a8-e42519529e6a",
            "webhookUsed": "false",
            "webhookForSlotFillingUsed": "false",
            "isFallbackIntent": "false",
```

```
            "intentName": "avere-caldo"
        },
        "fulfillment": {
            "speech": "",
            "messages": [
                {
                    "type": 0,
                    "speech": ""
                }
            ]
        },
        "score": 0.46000000834465027
    },
    "status": {
        "code": 200,
        "errorType": "success"
    },
    "sessionId": "123456789"
}
```

Bibliografia

[1] J. Austin, *How to Do Things with Words: The William James Lectures delivered at Harvard University in 1955*, Clarendon Press, Oxford, 1962

[2] J. Brownlee, on MachineLearningMastery.com, *A Gentle Introduction to k-fold Cross-Validation*, 2018, [Online, accesso: 05-07-2018].

[3] H. Bunt, W. Black in "Abduction, Belief and Context in Dialogue: Studies in computational pragmatics", *The ABC of Computational Pragmatics*,?John Benjamins Publishing Company, Amsterdam/Philadelphia, 2000

[4] F. Cady, in "The Data Science Handbook"', *Chapter 16 - Natural Language Processing*, John Wiley & Sons, 2017.

[5] J. Couto, on TryoLabs.com, *Building a Chatbot: analysis & limitations of modern platforms*, 2017, [Online, accesso: 05-07-2018].

[6] F. Domaneschi, C. Penco, "Presupposizioni", in APhEx 15, 2017.

[7] H. Grice, in P. Cole, J. Morgan "Syntax and Semantics, 3: Speech Acts", *Logic and conversation*, Academic Press, New York, 1975

[8] G. James, D. Witten, T. Hastie, R. Tibshirani, *An Introduction to Statistical Learning (with Applications in R)*, Springer, 2013

[9] M. Juan, M. Torres, in "Automatic Text Summarization"', *Appendix 1 - Information Retrieval, NLP and Automatic Text Summarization*, John Wiley & Sons, 2014.

[10] D. Jurafsky, J. Martin, *Speech and Language Processing (3rd ed. draft)*, Bozza del 20-08-2017

[11] R. Kohavi, F. Provost, in "Machine Learning (1998) 30: 271"', *Glossary of Terms*, Kluwer Academic Publishers, 1998

[12] M. Kuhn,K. Johnson, *Applied Predictive Modeling*, Springer, 2013

[13] Lhussiez P. 'Depado', on Blog.depado.eu, *DialogFlow: A complete guide with webhook*, 2018, [Online, accesso: 05-07-2018].

[14] K. Markham, on DataSchool.io, *Simple guide to confusion matrix terminology*, 2014, [Online, accesso: 27-07-2018]

[15] D. Poole, A. Mackworth, *Artificial Intelligence: Foundations of Computational Agents (2nd Edition)*, Cambridge University Press, 2017

[16] S. Raschka, V. Mirjalili, *Python Machine Learning*, Packt, 2017