Chatbot e Vocal Assistant: Nuove frontiere della comunicazione

Cosa sono i chatbot? Come i chatbot possono migliorare l'assistenza clienti? Cosa c'è dietro il funzionamento di un chatbot? In questo articolo si cercherà di dare una risposta a queste domande, fornendo anche un esempio pratico di come un software di dialogo automatizzato possa integrarsi in un contesto reale.

Introduzione

Un chatbot è un software atto ad emulare una conversazione, quanto più fluida e naturale, con un essere umano.

Il Deep Learning, come per ogni campo, è intervenuto anche qua rivoluzionando il modo di costruire agenti conversazionali automatizzati. Se prima esistevano solo dei semplici software che tramite regole fisse rispondevano solo ad un certo numero di domande formulate solamente in determinati modi, ora possiamo fruire di chatbot capaci di interpretare il linguaggio umano in maniera quasi del tutto naturale e trasparente

Molte volte questi ultimi fanno uso anche di moduli aggiuntivi di Text-To-Speech o Speech-To-Text per rendere l'interazione tra umano e macchina ancora più semplice e intuitiva. Altre volte invece è possibile aggiungere dei moduli basati su CNN[1] per il riconoscimento e descrizione di immagini.

Tradizionalmente i chatbot sono suddivisi in due categorie: a dominio chiuso e dominio aperto. I primi hanno lo scopo di rispondere a domande circoscritte, per l'appunto, ad un dominio ristretto. Un esempio può essere il chatbot usato per raccogliere ordinazioni o prenotazioni, per realizzare sistemi di assistenza e via discorrendo. Sono degli agenti che per loro natura risultano essere molto robusti in quanto devono operare solamente in pochissimi campi e casistiche rendendo così la gestione di eventuali input inaspettati molto basilare. Il secondo tipo di agenti, quelli cosiddetti a dominio aperto, tentano di imitare nella maniera più fedele possibile una conversazione con un essere umano. Attualmente i tre modelli che raggiungono prestazioni migliori sono Meena di Google, Blender di Facebook e GPT-3 di OpenAI. Recentemente alcuni lavori hanno tentato di unire questi due mondi realizzando alcuni chatbot capaci sia di fare conversazione sia di rispondere a domande specifiche all'ambiente per cui sono stati progettati. (Generative Encoder-Decoder Models for Task-Oriented Spoken Dialog Systems with Chatting Capability, 2017)

Una possibile applicazione pratica

L'esempio più classico per un chatbot è quello realizzato ai fini del Customer Service. Più in particolare, in un lavoro recente (A Financial Service Chatbot based on Deep Bidirectional Transformers, 2020), è introdotto un chatbot chiamato "*AVA: A Vanguard Assistant*" appositamente nato per l'assistenza nel settore finanziario. Essendo un chatbot realizzato ai fini di realizzare customer service è ovviamente un chatbot task oriented, appartenente alla categoria di chatbot a dominio chiuso, che è utilizzato all'interno del call center. Lo schema di funzionamento è elementare: durante una telefonata, l'operatore interagisce con il cliente e finché ne ha conoscenza risponde a tutti i quesiti che gli sono stati sottoposti. Durante la telefonata AVA è in ascolto e, quando l'operatore è in difficoltà, interroga una repository interna di

[1] https://www.agrifood.tech/analisti-ed-esperti/smart-agriculture-deep-learning/

documenti per fornire un'assistenza rapida. In Figura 1 è presente lo schema di funzionamento.

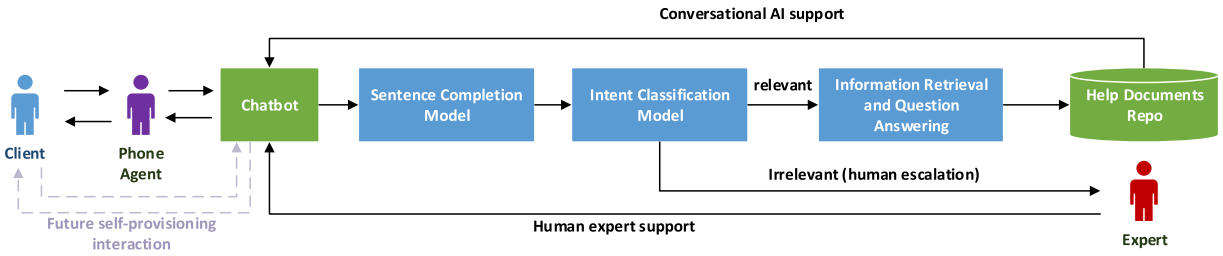

Figura 1: Schema di funzionamento di AVA

Il vantaggio è sicuramente marcato in quanto tradizionalmente, nel caso in cui l'operatore non conosce la risposta ad una domanda deve interrogare l'esperto mettendo in attesa il cliente. Un Chatbot, rispetto ad un essere umano, è sicuramente più veloce a fornire una risposta e sempre disponibile per l'utilizzatore. Tutto ciò porta l'assistenza clienti su un altro livello rendendola molto più veloce, economica e fluida. In altre parole, tramite Deep Learning, si rende più efficiente un processo che tradizionalmente è fermo da anni. La domanda a questo punto sorge spontanea: come funziona AVA? Il core di questo lavoro è un modello di deep learning, rilasciato da Google, chiamato BERT che verrà approfondito nel prossimo paragrafo. Per maggiori dettagli su come è stato utilizzato BERT e sul tipo di domanda a cui può rispondere un chatbot del genere invece si fa riferimento all'articolo completo (A Financial Service Chatbot based on Deep Bidirectional Transformers, 2020) e alla repository GitHub del progetto cyberyu/ava (github.com).

BERT

BERT, anagramma di *Bidirectional Encoder Representation from Transformers*, è un nuovo modello di rappresentazione del linguaggio progettato dai ricercatori di Google. La potenza di BERT è riassumibile fondamentalmente in due punti cardine: il primo riguarda l'addestramento, il secondo il meccanismo di self-attention. L'addestramento di BERT procede in due fasi, ovvero una di pre-training effettuata in maniera non supervisionata ed una di fine tuning per adattarlo al task specifico. Brevemente l'addestramento si basa su un principio chiamato *"Masked Language Model"* nel quale si maschera un token (parola) della frase per poi effettuare la predizione di quest'ultimo. Il secondo punto cardine riguarda intrinsecamente il modo in cui BERT è stato pensato in quanto, a differenza di molti altri modelli come possono essere GPT[2] o XLMNet, opera in maniera bidirezionale sull'input scorrendolo da destra a sinistra e da sinistra a destra; così facendo riesce a dare importanza a ciò che precede e sussegue una singola parola. Questi aspetti, per ora volutamente accennati, verranno successivamente trattati in maniera più dettagliata. BERT, quindi, risulta essere un modello molto generico adattabile a molteplici situazioni con un effort minimo e capace di raggiungere prestazioni esaltanti in task come il Question Answering, la classificazione di frasi e via discorrendo.

Struttura di BERT

La struttura interna di BERT è formata da diversi layer di *Transformers*: una particola struttura di tipo encoder-decoder che fa uso del meccanismo della self-attention per migliorare drasticamente le performance. Lo scopo di un encoder è quello, dato in input una sequenza di simboli nello spazio discreto, rappresentare questi simboli in uno spazio continuo. In maniera analogamente inversa invece lo scopo di un decoder è di trasformare questa rappresentazione continua di nuovo in una rappresentazione di simboli in uno spazio discreto. Come detto in precedenza BERT è un modello per rappresentare un linguaggio quindi, di Transformers, si utilizza solamente la parte di encoding escludendo i decoder. Prendendo in

[2] GPT e XLMNet sono due modelli, rilasciati rispettivamente da OpenAI e Facebook, basati anch'essi sui layer Transformer.

esame solamente vanilla BERT ne esistono due varianti: BERT Base e BERT Large differenziate esclusivamente dal numero di layer di encoder.

Affrontiamo ora la struttura di un singolo layer di encoding. Brevemente l'input di BERT è una o più frasi; una frase è composta da parole ed esse singolarmente, tramite una tecnica detta di Word Embedding (BERT in particolare usa WordPiece (Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation, 2016)), sono tramutate in più vettori di dimensione fissata (x i). Questi vettori entreranno all'interno del primo encoder, formato da due parti. La prima parte è il meccanismo di Self Attention che restituirà altrettanti vettori (z i) che andranno in input ad una FeedForward Neural Network il cui output sarà a sua volta altri tre vettori (r i). L'output finale a sua volta andrà in input agli altri encoder fino a raggiungere l'ultimo encoder. Gli encoder in totale sono 12 per BERT base e 24 per BERT Large.

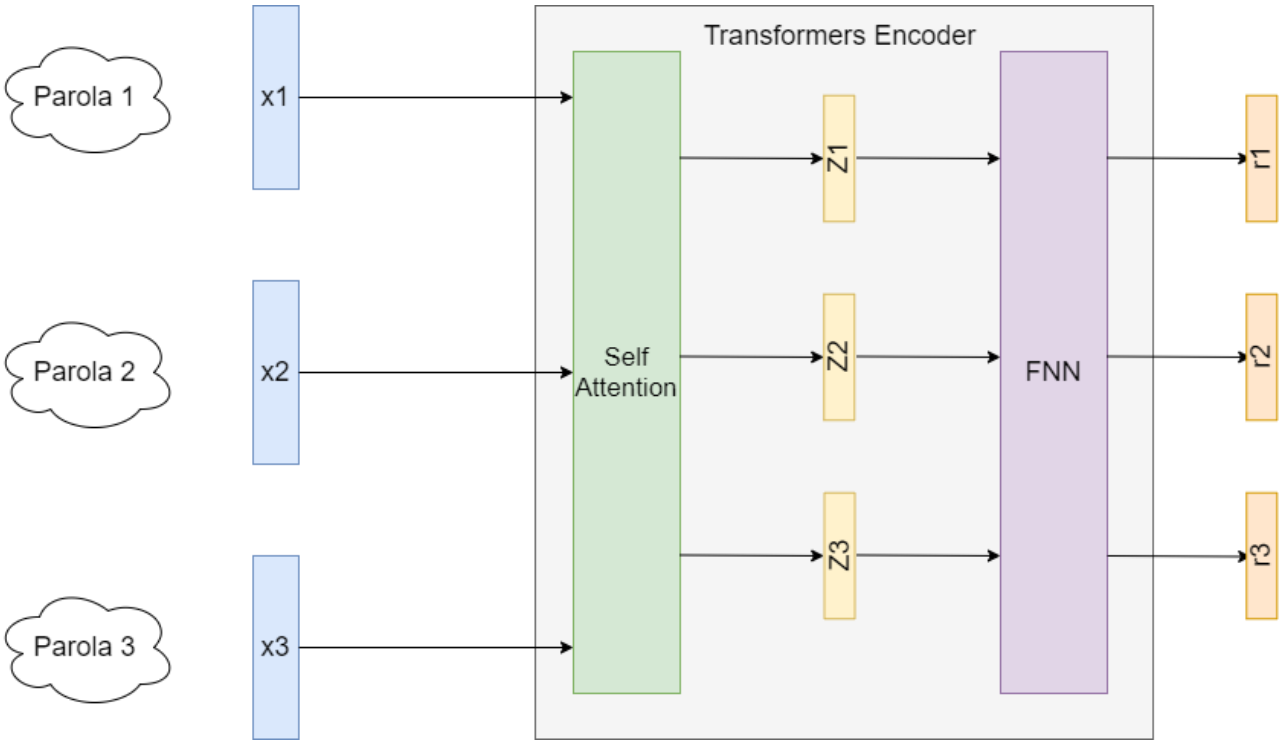

Figura 2: Schema del layer dell'encoder usato in BERT.

La particolarità però è la componente di Self Attention molto all'avanguardia. Tramite la definizione di tre matrici, chiamate **Q**uery, **K**ey e **V**alue, e la variazione dei loro valori durante la fase di addestramento si riesce a contestualizzare e comprendere al meglio una frase. Per maggiori dettagli, si rimanda al paper completo (Attention Is All You Need, 2017). A questo punto però la domanda sorge spontanea: perché se a monte c'è già un meccanismo che permette di trasformare parole in vettori è necessario fare tutto questo processamento? La risposta è più semplice di quanto possa sembrare. Prendiamo ad esempio un caso limite. La frase "La pesca è caduta dall'albero" e la frase "Sono andato a pesca di trote" hanno in comune una parola, ovvero -pesca-, che però ha due significati diametralmente differenti. Un sistema che tiene conto solamente della singola parola per trasformarla in una rappresentazione digeribile da una macchina non coglie questa differenza essenziale, BERT sì in quanto tiene conto del contesto attorno ad una parola.

Addestramento di BERT

L'addestramento di BERT procede in due fasi: la prima di pre-training e la seconda di fine-tuning. Nella prima frase è un po' come se si cercasse di far apprendere al modello la lingua su cui andrà ad operare. Si fa quindi un addestramento non supervisionato su quanto più testo possibile e si procede in due ulteriori fasi. Durante la prima parte del pretraining si usa una procedura detta *"Masked Language Model"* nel quale con

una certa probabilità si maschera una o più parole di una frase per poi effettuare una predizione. La seconda fase invece viene detta *"Next Sentence Prediction"* nel quale date due frasi il modello effettua una classificazione binaria su di esse. Le classi oggetto della predizione sono "*isNext"* e *"notNext"* e rappresentano rispettivamente la possibilità, o meno, che la seconda frase sia il proseguo della prima.

La fase di fine tuning di BERT è invece molto basilare e serve a adattare il modello al task specifico che andrà a compiere. Ad esempio, per realizzare un sistema di question answering, è sufficiente fornire a BERT coppie domanda-risposta. Per maggiori dettagli sulla fase di training di BERT si rimanda all'articolo completo (BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018).

Conclusioni

Il Deep Learning può rivoluzionare completamente l'assistenza clienti e rendere molto più agevole la conversazione di un operatore telefonico e un assistito. In particolare, l'attività può essere più efficiente in termini di:

- Costi: le interazioni più semplici possono essere delegate ai chatbot e assistenti vocali, lasciando l'operatore concentrato sulle interazioni a maggior valore aggiunto;
- Efficacia: gli strumenti di intelligenza artificiale possono reperire molte informazioni e metterle a disposizione dell'operatore e dell'assistito;
- Velocità: i tempi di attesa possono allungarsi notevolmente in determinate fasce della giornata o durante particolari eventi. I chatbot e i vocal assistant possono alleggerire il carico evadendo le richieste più semplici e svolgendo compiti "standard" come la raccolta dei dati preliminari a una richiesta.

D'altra parte, si ritrovano un po' tutti quegli svantaggi che si hanno quando si opera con un sistema di deep learning, in particolare una massiva richiesta di dati e la necessità di hardware (computer e schede video) per addestrare i modelli.

Pro:

- Costi di esercizio minori
- Velocità
- Disponibilità 24/7

Contro:

- Hardware costoso
- Raccolta dati

Riferimenti

A Financial Service Chatbot based on Deep Bidirectional Transformers. **Shi Yu, Yuxin Chen, Hussain Zaidi. 2020.** 2020.

Attention Is All You Need. **Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. 2017.** 2017.

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. **Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. 2018.** 2018.

Generative Encoder-Decoder Models for Task-Oriented Spoken Dialog Systems with Chatting Capability. **Tiancheng Zhao, Allen Lu, Kyusong Lee and Maxine Eskenazi. 2017.** 2017.

Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. **Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, H. 2016.** 2016.