



UNIVERSITÀ DEGLI STUDI DI SALERNO

Dipartimento di Informatica

Corso di Laurea Triennale in Informatica

TESI DI LAUREA

# **Algoritmi di Intelligenza artificiale per lo sviluppo di un layer analitico per l'interpretazione fenomenologica ambientale**

RELATORE

Prof. Fabio Palomba

Ing. Daniele Sofia

Università degli Studi di Salerno

CANDIDATO

**Antonio Maddaloni**

Matricola: 0512107890

Anno Accademico 2021-2022

*Questa tesi è stata realizzata nel*

sesa<sup>lab</sup>  
SOFTWARE ENGINEERING  
SALERNO

*A te nonna,  
anche se oggi non sei qui con noi, so che saresti fiera di me.*

## Sommario

La tesi presentata offre una panoramica generale del mondo del Natural Language Processing, delle sue principali attività e delle branche ad essa associate. Di conseguenza si mostra l'utilizzo di tecniche di Deep Learning per risolvere problematiche relative all'NLP, le quali in poco tempo si sono evolute, soprattutto per i modelli di linguaggio. Tutto questo ci ha permesso di introdurre l'argomento centrale dell'elaborato, ovvero l'utilizzo di GPT-2 per la modellazione e la generazione di report ambientali. Vengono così presentate una porzione delle attività proposte dalla text generation, che hanno permesso di raggiungere lo stato dell'arte. Tra le varie tecniche, verrà analizzata principalmente quella dei Transformer, presentando le varie tipologie presenti, utile per introdursi verso l'utilizzo di GPT-2. Attraverso GPT-2, si è realizzato un sistema di intelligenza artificiale che fosse in grado di generare report ambientali, seguendo una pipeline ben definita. Partendo dalla costruzione del dataset, con il successivo processo di data cleaning, proseguendo poi con la fase di addestramento ed infine la generazione del report. Tale sistema si è dimostrato abbastanza soddisfacente per uno sviluppo iniziale, che farà da base a sviluppi futuri.

---

## Indice

---

<b>Indice</b>	<b>i</b>
<b>Elenco delle Figure</b>	<b>iii</b>
<b>Elenco delle Tabelle</b>	<b>v</b>
<b>1 Introduzione</b>	<b>1</b>
1.1 Sense Square e la qualità dell'aria . . . . .	1
1.2 Motivazioni e obiettivi . . . . .	4
1.3 Risultati ottenuti . . . . .	4
1.4 Struttura della tesi . . . . .	5
<b>2 Background e Stato dell'arte</b>	<b>6</b>
2.1 Natural Language Processing . . . . .	6
2.1.1 Natural Language Understanding . . . . .	7
2.1.2 Natural Language Generation . . . . .	8
2.2 Deep Learning . . . . .	9
2.2.1 Reti Neurali . . . . .	12
2.2.2 Vantaggi e difficoltà dell'utilizzo del Deep Learning . . . . .	16
2.3 Modelli di linguaggio per la generazione del testo . . . . .	17
2.3.1 Modello linguistico basato su n-grams . . . . .	18

---

2.3.2	Modelli linguistici basati su Deep Learning . . . . .	19
2.4	GPT-2 . . . . .	25
2.4.1	Architettura e funzionamento . . . . .	27
<b>3</b>	<b>Implementazione</b>	<b>35</b>
3.1	Specifiche tecniche e librerie utilizzate . . . . .	36
3.2	Costruzione del dataset . . . . .	38
3.3	Tecniche di Data Cleaning e traduzione del dataset . . . . .	42
3.3.1	Traduzione del dataset . . . . .	42
3.3.2	Sentence tokenization . . . . .	44
3.3.3	Lowercase . . . . .	45
3.3.4	Rimozione punteggiatura . . . . .	47
3.3.5	Word tokenization . . . . .	48
3.3.6	Rimozione dati rumorosi . . . . .	50
3.4	Addestramento . . . . .	51
3.5	Generazione report . . . . .	52
<b>4</b>	<b>Valutazione del modello</b>	<b>54</b>
4.1	Valutazione del dataset ottenuto . . . . .	55
4.2	Valutazione delle frasi generate . . . . .	57
<b>5</b>	<b>Conclusioni e sviluppi futuri</b>	<b>62</b>
	<b>Bibliografia</b>	<b>64</b>

---

## Elenco delle figure

---

1.1	Piattaforma per il monitoraggio della qualità dell'aria.[1]	2
1.2	Livello di inquinamento per la regione Campania.[1]	2
1.3	Monitoraggio giornaliero degli inquinanti.[1]	3
2.1	Suddivisione dell'NLP. [2]	7
2.2	Branche dell'AI. [3]	11
2.3	Neurone. [4]	12
2.4	Neurone all'interno di una rete neurale artificiale. [5]	13
2.5	Rete neurale artificiale. [6]	13
2.6	Funzione di attivazione. [7]	14
2.7	Recurrent Neural Network (RNN). [8]	22
2.8	Long Short-Term Memory (LSTM). [9]	23
2.9	Rappresentazione di una rete Transformer. [10]	24
2.10	Modelli GPT, BERT. [11]	25
2.11	Varianti del modello di GPT-2. [12]	26
2.12	Architettura transformer. [12]	27
2.13	Modelli GPT, BERT, TRANSFORMER XL. [12]	27
2.14	Blocco Encoder. [12]	28
2.15	Blocco Decoder. [12]	28
2.16	Generazione token(1). [12]	29

2.17	Generazione token(2). [12]	30
2.18	Generazione token(3). [12]	30
2.19	Differenze tra Self-Attention e Masked Self-Attention. [12]	31
2.20	Architettura GPT-2. [12]	31
3.1	Pipeline implementazione.	35
3.2	Dataset ottenuto tramite l'estrazione.	41
3.3	Dataset tradotto.	43
3.4	Dataset dopo aver effettuato la sentence tokenization.	45
3.5	Dataset dopo la trasformazione dei dati testuali in minuscolo.	46
3.6	Dataset dopo aver rimosso la punteggiatura.	48
3.7	Dataset dopo aver effettuato la Word tokenization.	49
4.1	Word Cloud del dataset iniziale.	55
4.2	Word Cloud del dataset finale.	56
4.3	prefisso=Today in Milan, Lunghezza=100.	57
4.4	prefisso= Naples, Lunghezza=100.	58
4.5	prefisso= The aqi, Lunghezza=150.	58
4.6	prefisso= The aqi, Lunghezza=200.	58
4.7	prefisso= The aqi, Lunghezza=200.	59



---

## Elenco delle tabelle

---

3.1	Differenze delle versioni fornite da GPT-2-SIMPLE [13]. . . . .	36
4.1	Parametri per la generazione di frasi . . . . .	57
4.2	Scala di leggibilità . . . . .	60
4.3	Risultati delle frasi ottenute tramite l'indice di Gulpease . . . . .	60

# CAPITOLO 1

---

## Introduzione

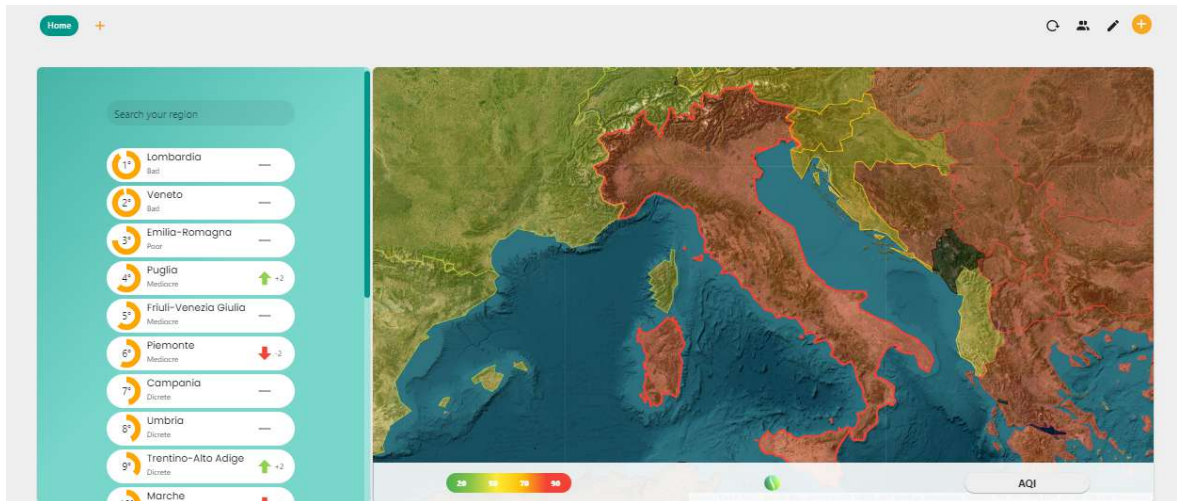
---

### 1.1 Sense Square e la qualità dell'aria

L'aria è un miscuglio di sostanze aeriformi che costituisce l'atmosfera terrestre. È indispensabile per la vita della maggior parte degli organismi viventi, in particolare per l'uomo poiché necessaria per la respirazione. Oltre all'ossigeno necessario, ad ogni respiro, inaliamo anche piccole quantità di gas potenzialmente dannosi. Questi componenti influenzano direttamente la nostra salute e dunque, con il passare del tempo, l'uomo ha iniziato a preoccuparsi per la qualità dell'aria. Quando si parla di qualità dell'aria ci si riferisce alla presenza di inquinanti nell'aria ambiente. La presenza di quantitativi significativi di inquinanti, rispetto ai valori limite individuati dalla legge, pregiudica la qualità dell'aria. Esistono diverse aziende che si occupano del monitoraggio della qualità dell'aria, in particolare Sense Square.

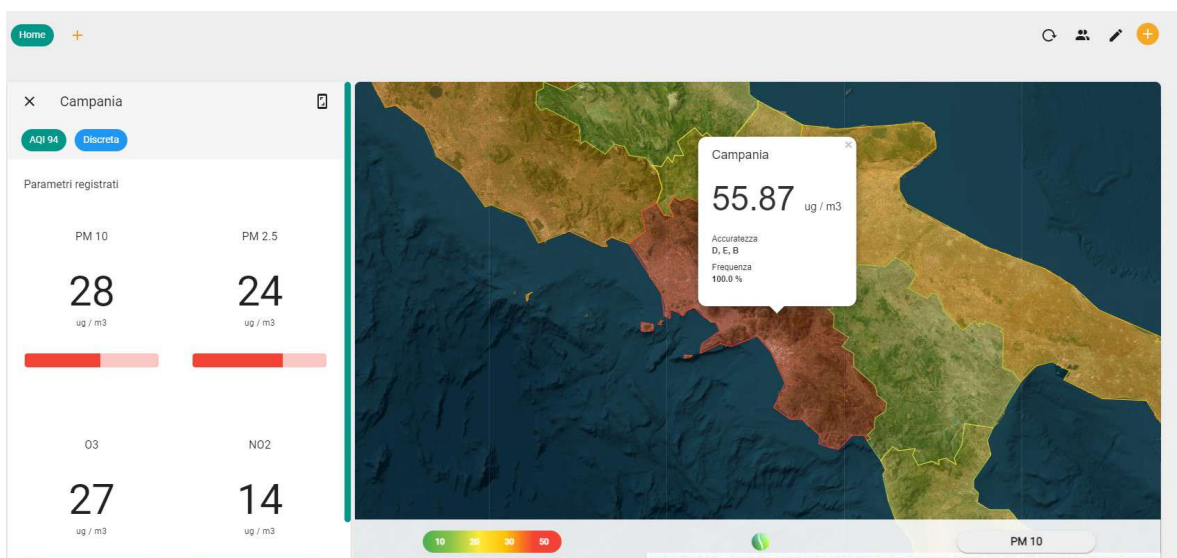
Sense Square Srl è una start-up innovativa specializzata nella creazione di reti tecnologiche per la raccolta e aggregazione di dati territoriali georeferenziati. Nello specifico, l'azienda ha sviluppato un sistema automatico di tracciamento delle sorgenti di inquinamento. Il fine della start-up è quello di rendere il monitoraggio della qualità dell'aria un diritto di tutti i cittadini (attraverso una piattaforma Fig. 1.1), così

da renderli consapevoli dell'aria che respirano, permettendogli, così, di assumere comportamenti ed intraprendere azioni per limitare gli impatti negativi sulla propria salute.



**Figura 1.1:** Piattaforma per il monitoraggio della qualità dell'aria.[1]

Con i dati a disposizione, l'azienda Sense Square può analizzare e monitorare la qualità dell'aria a diversi livelli di dettaglio, partendo da intere nazioni fino ai singoli km<sup>2</sup> (Fig.1.2).

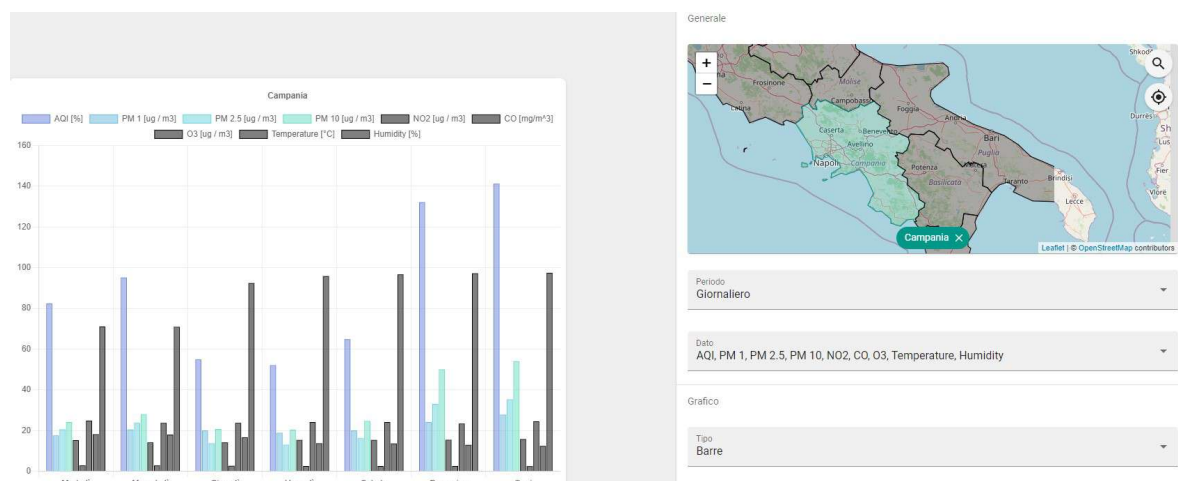


**Figura 1.2:** Livello di inquinamento per la regione Campania.[1]

Grazie a tale piattaforma, Sense Square riesce a monitorare e visualizzare le percentuali di diffusione dei principali inquinanti dell'aria:

- AQI
- PM1
- PM2,5
- PM10
- VOC
- NO2
- CO
- O3
- SO2
- H2S

Tali percentuali possono essere distribuite in base alla media giornaliera, alla media oraria e alla media storica. Questa distribuzione consente di poter avere un controllo completo anche grazie alla possibilità di poter scegliere tra diversi tipi di grafici (come, ad esempio, il grafico a barre, linea...) (Fig.1.3).



**Figura 1.3:** Monitoraggio giornaliero degli inquinanti.[1]

## 1.2 Motivazioni e obiettivi

Per migliorare ulteriormente il monitoraggio della qualità dell'aria, in modo da poter rendere i cittadini sempre più consapevoli delle condizioni ambientali, sulla piattaforma di Sense Square, è nata la necessità di sviluppare un sistema del tutto automatico ed intelligente. Il sistema proposto deve essere in grado di generare dei piccoli report ambientali, che descrivono in maniera accurata lo stato della qualità dell'aria ed eventi storici di una determinata provincia o regione, partendo da dati reali o proprietari. Quindi le motivazioni della stesura del seguente elaborato, sono quelle di realizzare un sistema di Deep Learning che soddisfi gli obiettivi preposti da Sense Square.

## 1.3 Risultati ottenuti

Attraverso l'uso di GPT-2, si ottiene un quadro completo sulle moderne tecniche di Intelligenza Artificiale, che tutt'oggi sono in continua evoluzione. Infatti, si sa che GPT-2 non è l'ultima versione della serie GPT, ma attualmente, risulta essere l'ultima della serie che sia più aggiornata (rispetto ai suoi predecessori) e open source, che fa al caso di una start-up. I risultati ottenuti non vogliono aprire nuovi orizzonti a differenti approcci sullo studio, ma fanno più da panoramica generale a tecniche già esistenti, che a loro volta, fanno da base ad un sistema che dovrà essere più complesso in futuro. Infine, è stato sviluppato un sistema di intelligenza artificiale per la generazione di report ambientali, addestrato su un dataset ottenuto grazie ad un processo di data cleaning, in grado di fornire informazioni inerenti al contesto. Tale sistema è stato valutato in primis in maniera empirica tramite una valutazione fornita da esperti del dominio, in modo da valutarne la correttezza sintattica e la coerenza semantica. Infine, si è fatto uso di un indice di leggibilità chiamato Gulpease, in modo da poter avere un risultato numerico del grado di comprensione della frase, al fine di ricavarne le prestazioni e possibili miglioramenti, che contribuiranno al miglioramento del modello in futuro.

## 1.4 Struttura della tesi

Il seguente elaborato è strutturato in vari capitoli, ognuno con lo scopo di presentare e descrivere in modo approfondito diversi aspetti della ricerca effettuata e del modello sviluppato. Nel Capitolo 1 sono elencati gli obiettivi e le motivazioni relative all'esigenza di dover costruire un modello per un miglioramento del monitoraggio dei dati. Nel Capitolo 2 è analizzato il Background e lo Stato dell'Arte, ossia ci si concentra sul NLP in generale da un punto di vista puramente teorico, approfondendo una particolare branca di NLP, denominata NLG. Si parte dai primi modelli di generazione del testo, arrivando infine alle ultime architetture di rete usate nell'ambito del Natural Language Generation, nello specifico quelle della serie GPT. Nel Capitolo 3 viene presentata la pipeline di Machine Learning impiegata per l'addestramento del modello per la generazione dei report ambientali, presentando le tecnologie utilizzate, le varie tecniche di Pre-Processing applicate ai dati e il possibile utilizzo del modello. Nel Capitolo 4 saranno descritti tutti i risultati che sono stati ottenuti, cercando di valutarli in maniera empirica. Infine, nel Capitolo 5 saranno presentate le conclusioni ed eventuali sviluppi per lavori futuri.

---

### Background e Stato dell'arte

---

#### 2.1 Natural Language Processing

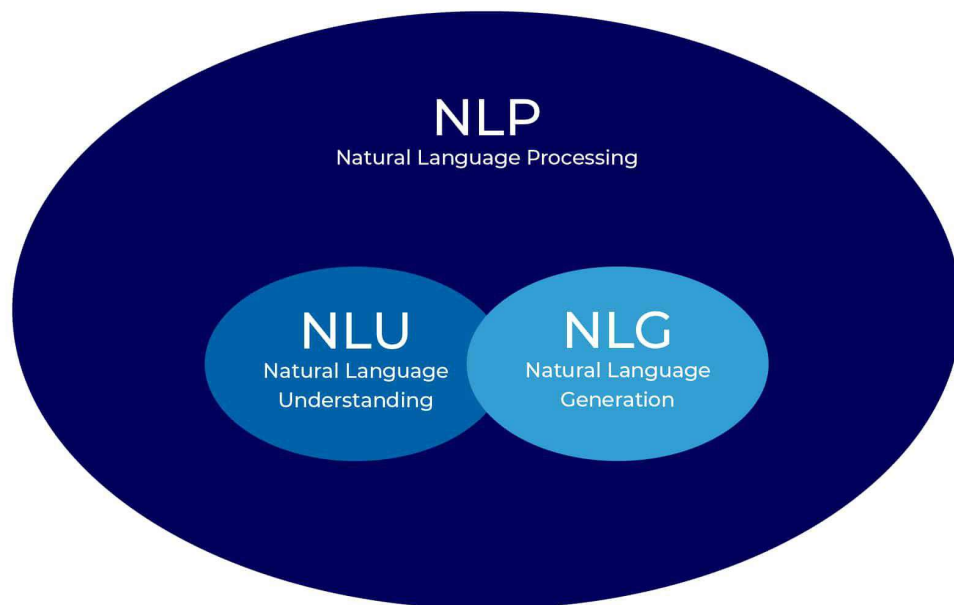
Il Natural Language Processing è una branca della computer science e della linguistica. Si occupa di fornire ai sistemi intelligenti la possibilità di leggere e capire il linguaggio utilizzato dagli esseri umani (come, ad esempio, un testo scritto o una conversazione orale) in una maniera che sia produttiva ed efficiente [14]. La difficoltà principale di questo processo è l'intrinseca ambiguità che caratterizza i linguaggi naturali, per questo motivo le soluzioni richiedono un'estesa conoscenza e una notevole abilità nel manipolarla. Infatti, si deve tener conto della struttura, della grammatica e di un lessico adeguato al contesto. Per tener conto di ciò, si utilizza una fase di pre-processing, di fatti, nel Natural Language Processing, esistono due fasi principali: data pre-processing e lo sviluppo di algoritmi. La fase di data pre-processing consiste nel preparare e pulire i dati testuali per renderli analizzabili dalla macchina, in modo tale che essi siano in una forma lavorabile [15]. Ci sono molti modi con cui può essere fatto, tra i quali: [16]

- Tokenization: che si occupa di suddividere il testo in unità minime di analisi (chiamate token). La suddivisione in token rende possibili gli ulteriori step di pre-processing

che, altrimenti, risulterebbero complessi o inefficaci. Esistono due tipi di Tokenization: La sentence tokenization, che consiste di avere come unità minima di analisi un'intera frase e la word tokenization, che suddivide l'intero testo in parole.

- La rimozione dei “noisy data”, ovvero eliminare i dati rumorosi (ad esempio un link...), in modo tale che vengano lasciate nel testo solo i dati rilevanti.
- Lemmatization e stemming, per ridurre le parole alla loro radice.

Infine, l'intera branca dell'NLP si suddivide in due categorie, NLG e NLU. [14]



**Figura 2.1:** Suddivisione dell'NLP. [2]

### 2.1.1 Natural Language Understanding

Lo scopo dell'NLU (Natural Language Understanding) riguarda la capacità delle macchine di comprendere e utilizzare l'analisi sintattica e semantica per determinare il significato di un testo. Esso può essere applicato ad una serie di processi, come la classificazione dei testi, la raccolta di informazioni e l'analisi dei contenuti. Il Natural Language Understanding è un campo in evoluzione e mutevole ed è considerato uno dei problemi più difficili dell'IA. Sono in fase di sviluppo varie tecniche e strumenti per fornire alle macchine una comprensione del linguaggio umano [17].



Le varie tecniche includono: [17]

- Il riconoscimento di entità nominate: è il processo che opera distinguendo concetti e riferimenti fondamentali in un corpo di testo, identificando entità nominate e inserendole in categorie come luoghi, date, organizzazioni, persone, lavori, ecc.
- La disambiguazione del senso della parola: è il processo di determinazione del significato, o senso, di una parola in base al contesto in cui la parola appare.

Alcuni esempi comuni di NLU includono: ragionamento automatico, instradamento automatico dei ticket, traduzione automatica e risposta alle domande. [17]

### 2.1.2 Natural Language Generation

Per Natural Language Generation(NLG) intendiamo l'insieme di tecniche e algoritmi per la generazione automatica di informazioni scritte in linguaggio naturale, che è quello della lingua parlata. Nel caso applicativo di chatbot e assistenti vocali, esso fa parte dell'ultima fase del processo in cui viene compreso un input in linguaggio naturale, viene trasformato in un insieme di dati strutturati, ne viene elaborata una risposta o un'azione, infine l'output viene trasformato in un linguaggio simile a quello di input, che spesso è la lingua parlata e a volte tradotto in un'altra lingua per i sistemi più evoluti [18]. Tra i vantaggi nell'uso di NLG possiamo trovare: [18]

- Risparmio di tempo
- Creazione di contenuti in modo automatico, ovvero disporre di sistemi che producono contenuti testuali senza l'intervento umano.
- Scalabilità in tempo reale.

Tra le possibili applicazioni si può pensare a contesti in cui in sostituzione di un report o di una comunicazione schematica, si può produrre attraverso un documento nel linguaggio naturale più familiare e comprensibile; messaggi informativi nel settore ambientale, nel settore trasporti, nelle diverse branche della medicina, nella fruizione di contenuti web. Infine la necessità e la creatività determinano i limiti applicativi.

## 2.2 Deep Learning

Il deep learning è un metodo di intelligenza artificiale (IA) che insegna ai computer a elaborare i dati in un modo che si ispira al cervello umano. I modelli di deep learning sono in grado di riconoscere pattern complessi in immagini, testo, suoni e altri dati per produrre informazioni e previsioni accurate. Si possono utilizzare metodi di deep learning per automatizzare le attività, che in genere richiedono l'intelligenza umana, come la descrizione di immagini o la generazione di testo. Il deep learning è un campo specifico dell'apprendimento automatico (machine learning), che include modelli statistici e predittivi. Attraverso il deep learning si possono creare dei modelli di analisi predittiva, che apprendono da dati correnti e passati per predire attività, comportamenti e tendenze future. Gli algoritmi di deep learning sono emersi nel tentativo di rendere più efficienti le tecniche di machine learning tradizionali [19]. I metodi tradizionali di machine learning richiedono un notevole sforzo umano per addestrare il software. Ad esempio, nel riconoscimento delle immagini di animali, è necessario eseguire le seguenti operazioni: [19]

- Etichettare manualmente centinaia di migliaia di immagini di animali.
- Fare in modo che gli algoritmi di machine learning elaborino quelle immagini.
- Provare questi algoritmi su una serie di immagini sconosciute.
- Capire perché alcuni risultati non sono accurati.
- Migliorare il set di dati etichettando nuove immagini per migliorare l'accuratezza dei risultati.

Questo processo è chiamato apprendimento supervisionato. Nell'apprendimento supervisionato, l'accuratezza dei risultati migliora solo quando si dispone di un set di dati ampio e sufficientemente vario. Ad esempio, l'algoritmo potrebbe identificare accuratamente i gatti neri ma non i gatti bianchi perché il set di dati di addestramento conteneva più immagini di gatti neri. In tal caso, si dovrebbero etichettare più immagini di gatti bianchi e addestrare nuovamente i modelli di machine learning [19]. Esistono vari metodi per creare dei forti modelli di deep learning:

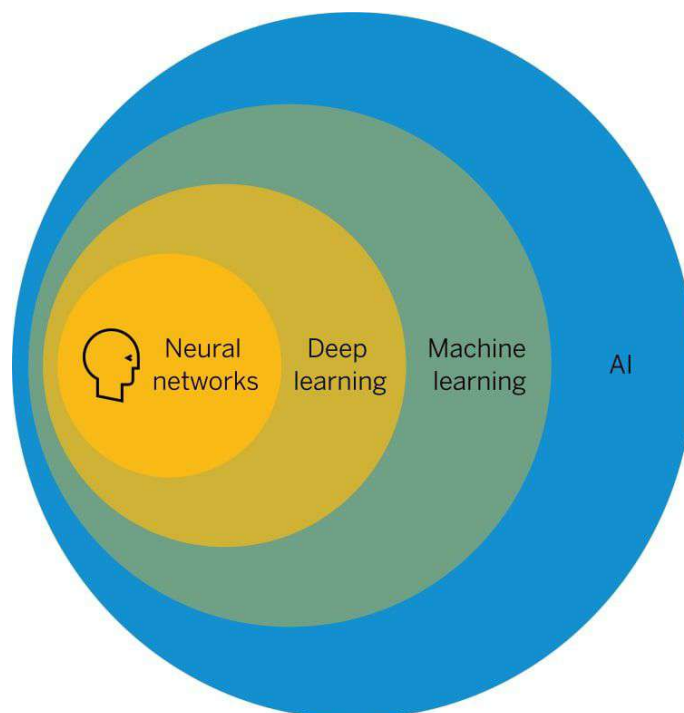
- **Transfer learning:** [20] Viene utilizzato per riaddestrare un modello di rete neurale, allo scopo di risolvere un problema simile a quello per cui è stato progettato, mediante l'utilizzo di uno o più livelli dello stesso. Il Transfer learning può essere molto utile nel caso in cui lo sviluppo del modello di rete neurale abbia portato a etichettare molti più dati rispetto al problema di iniziale interesse, e vi è una somiglianza nella struttura del nuovo problema da risolvere. L'obiettivo, dunque, è quello di sfruttare i dati della prima impostazione del modello neurale per estrarre informazioni che possono essere utili durante la fase di apprendimento del modello o per effettuare nuove previsioni in una seconda impostazione dello stesso.
- **Training from scratch:** [21] Consente agli sviluppatori di configurare da zero le architetture di rete raccogliendo grandi quantità di dati etichettati. Questo è il metodo meno utilizzato poiché l'addestramento richiede molto tempo a causa della grande quantità di dati. Rispetto ai metodi precedenti, richiede più dati e più tempo di elaborazione per ottenere risultati comparabili.
- **Dropout:** [22] Questa tecnica tenta di risolvere il problema dell'overfitting nelle reti con una grande quantità di dati, eliminando i neuroni dalla rete neurale durante l'addestramento, in altre parole, diversi neuroni vengono rimossi temporaneamente dalla rete. Durante l'addestramento, il dropout modifica l'idea di apprendere tutti i pesi nella rete, per apprendere solo una frazione dei pesi nella rete. Dopo ogni iterazione, vengono attivati diversi gruppi di neuroni, in modo da evitare che alcuni neuroni dominino il processo. Questo, quindi, ci aiuta a ridurre la minaccia dell'overfitting e consente la nascita di architetture di rete più profonde e più grandi che possono fare buone previsioni su dati.
- **Few-shot learning:** [23] Permette allo sviluppatore di creare un dataset con pochi esempi, in modo da eliminare il problema della grandezza del dataset. L'obiettivo principale nei framework Few-Shot tradizionali è quello di apprendere una funzione di somiglianza, che sia in grado di mappare le somiglianze tra le classi nei set di supporto e di dati. Le funzioni di somiglianza generano un valore di probabilità per la somiglianza.

Il deep learning ha diversi casi d'uso nel settore ambientale, automobilistico, aerospaziale, manifatturiero, elettronico, della ricerca medica e in altri campi [19]. Ecco

alcuni esempi di deep learning: [19]

- I sistemi per l'analisi e l'interpretazione fenomenologica ambientale.
- Le auto a guida autonoma utilizzano modelli di deep learning per rilevare automaticamente segnali stradali e pedoni.
- I sistemi di difesa utilizzano il deep learning per segnalare automaticamente le aree di interesse nelle immagini satellitari.
- L'analisi delle immagini mediche utilizza il deep learning per rilevare automaticamente le cellule tumorali per la diagnosi medica.
- Le fabbriche utilizzano applicazioni di deep learning per rilevare automaticamente quando persone o oggetti si trovano a una distanza non sicura dalle macchine.

Si possono raggruppare i diversi casi d'uso del deep learning in quattro grandi categorie: visione artificiale, riconoscimento vocale, elaborazione del linguaggio naturale (NLP) e motori di raccomandazione. [19]



**Figura 2.2:** Branche dell'AI. [3]

### 2.2.1 Reti Neurali

I modelli di Deep Learning sono formati da delle reti neurali artificiali, che non sono altro che dei modelli matematici che rappresentano lo stesso funzionamento di una rete neurale che si trova all'interno di un cervello umano [24]. Il termine rete neurale viene utilizzato come riferimento a una rete o a un circuito formato da neuroni, ovvero unità funzionali del sistema nervoso, un neurone è una cellula altamente specializzata per ricevere, elaborare e trasmettere le informazioni ad altri neuroni o altre cellule, attraverso segnali elettrici e chimici. Il neurone è costituito da quattro componenti: il corpo cellulare (pirenoforo o soma), i dendriti, il cono di emergenza e l'assone. [24, 25]

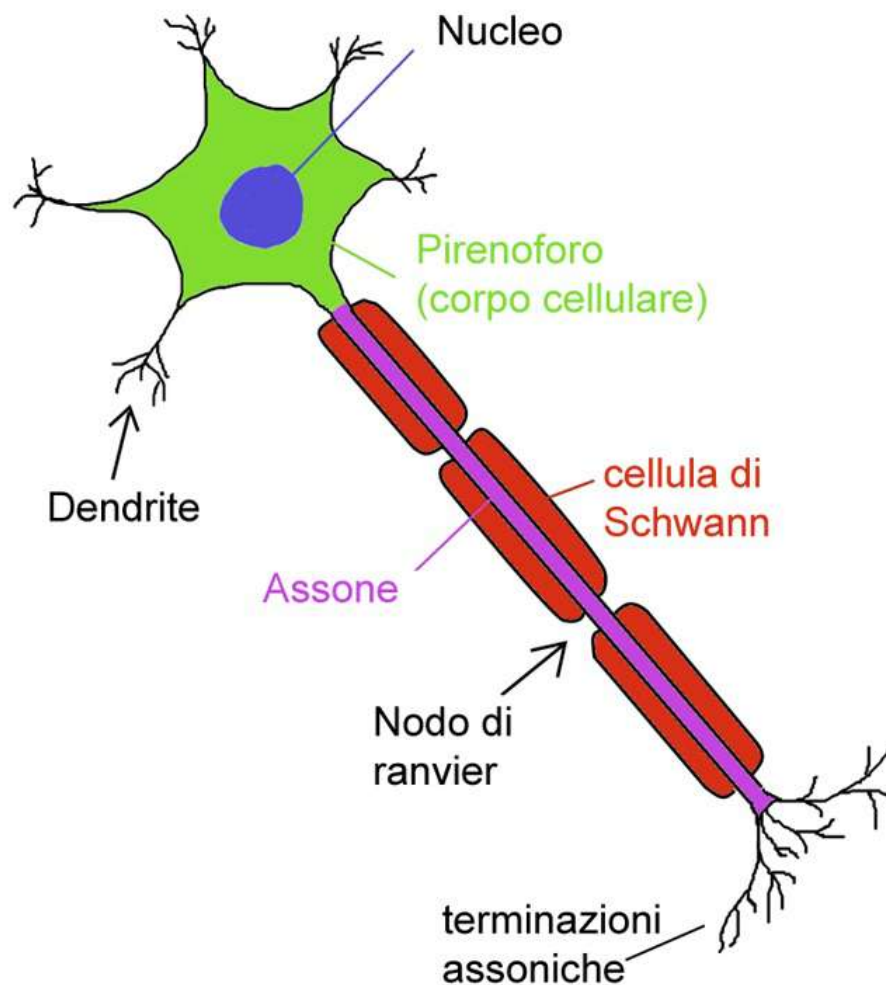
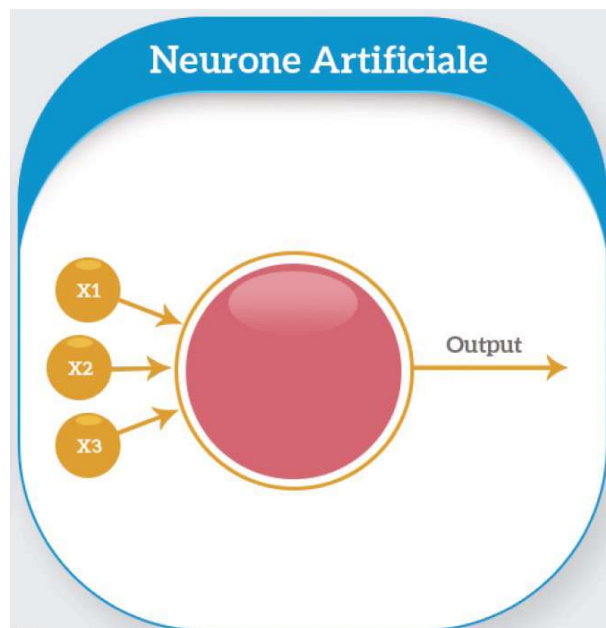


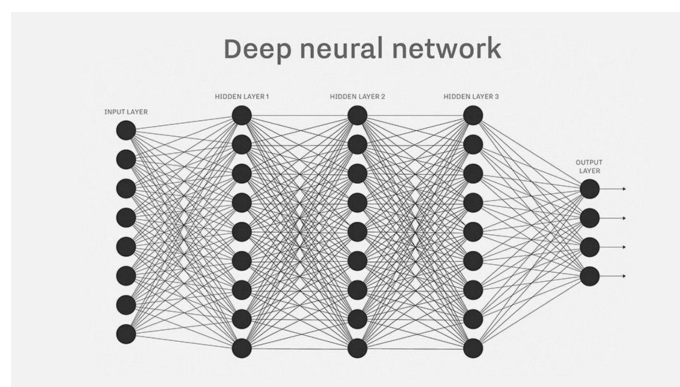
Figura 2.3: Neurone. [4]

In una rete neurale artificiale rappresentiamo un neurone nel seguente modo:



**Figura 2.4:** Neurone all'interno di una rete neurale artificiale. [5]

All'interno della rete neurale, ogni neurone ha dei collegamenti di input chiamati dendriti, associati a dei collegamenti di output (rappresentati da un singolo filamento chiamato assone). L'assone si divide, a sua volta, in diverse nervature che prendono il nome di terminali, che infine si collegheranno con i dendriti di un altro neurone [24]. Quindi otteniamo una rete neurale artificiale di questo tipo:



**Figura 2.5:** Rete neurale artificiale. [6]

I componenti di una rete neurale profonda sono i seguenti: [26, 19]

- Input layer

Una rete neurale artificiale ha diversi nodi che vi immettono dati. Questi nodi costituiscono il livello di input del sistema.

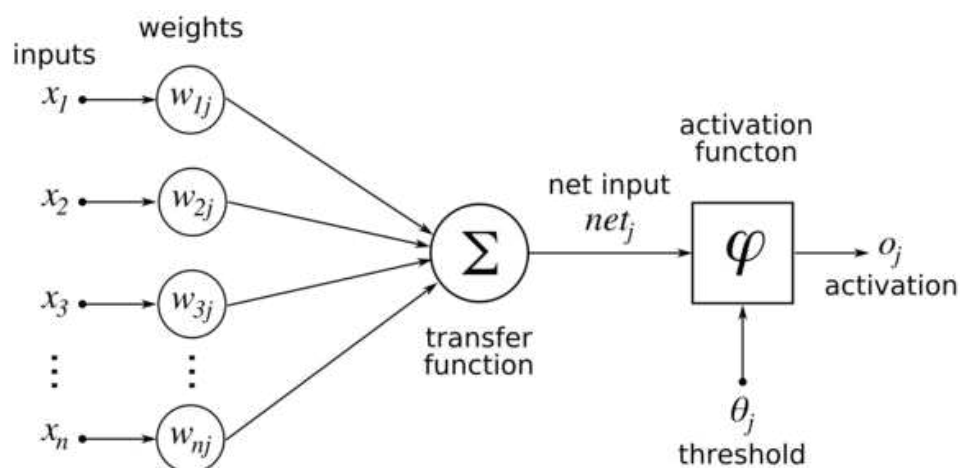
- Hidden layer

Il layer di input elabora e trasmette i dati ai livelli successivi nella rete neurale. Questi livelli nascosti (Hidden layer) elaborano le informazioni a diversi livelli, adattando il loro comportamento man mano che ricevono nuove informazioni. Le reti di deep learning hanno centinaia di livelli nascosti che possono utilizzare per analizzare un problema da diverse angolazioni.

- Output layer

Il livello di output è costituito dai nodi che generano i dati. I modelli di deep learning che generano risposte "sì" o "no" hanno solo due nodi nel livello di output. D'altra parte, quelli che producono una gamma più ampia di risposte hanno più nodi.

A differenza della programmazione "convenzionale", una rete neurale impara dai dati osservativi in una logica di "autoapprendimento", individuando la propria soluzione al problema. Prendendo in considerazione questa rete neurale:



**Figura 2.6:** Funzione di attivazione. [7]

Possiamo notare come ad ogni neurone, sia associata una funzione che ne determina il funzionamento [26] (prende il nome di funzione di trasferimento [27]). Ogni

input da in output un determinato valore, quindi, il neurone prende in input un  $Z$ , dove  $Z$  non è altro che la sommatoria pesata per ogni input  $x$  più un certo bias. Questo valore  $Z$  verrà dato in input ad una funzione di attivazione del neurone successivo. La funzione di attivazione non è altro che una funzione che determina l'attivazione o meno di un neurone, quindi, se in quell'iterata(epoca) l'input è rilevante o meno. Data una funzione di attivazione  $a(Z)$ , essa darà un output che chiameremo  $\hat{y}$ , quindi:

$$\hat{y} = a(Z)$$

Il valore di output sarà input di un altro neurone e così via. Esistono diversi tipi di funzione di attivazione e si cerca di avere una non linearità di essa e la migliore dipende dal contesto di utilizzo. Un esempio di funzione di attivazione è la SIGMOID. [26, 28]

Una rete neurale potrebbe restituire un output differente da quello atteso, per cui essendo basata su una logica di autoapprendimento, cercherà ad ogni epoca di migliorare il suo output in modo da avvicinarsi a quello atteso. Per fare ciò, ci si basa su una funzione di loss detta anche funzione di perdita. La funzione di loss si occupa di determinare l'errore che porta ad un risultato sbagliato della predizione, il tutto solo se si conosce il valore reale di ciò che si deve predire. Essa definisce di quanto l'output è errato in quella specifica iterazione. Una rete neurale in generale non si basa su un'unica iterazione, poiché nell'iterazione successiva l'errore potrebbe essere maggiore di quello precedente. Chi si occupa di tener conto dell'errore di ogni epoca è la funzione di costo. Esistono diversi tipi di funzione di costo e di loss. In base all'errore calcolato, attraverso le funzioni di costo e di loss, una rete neurale tenta di cambiare i pesi  $w$  e il Bias  $b$  di ogni neurone per ogni epoca, poiché si pone l'obiettivo di minimizzare l'errore nelle epoche successive. Il modo in cui la rete neurale tenta di cambiare i pesi  $w$  in funzione del costo viene calcolato attraverso delle derivate parziali, che in caso di miglioramento, cambierà i pesi  $w$  con i nuovi calcolati. La rete neurale si fermerà quando i pesi  $w$ , daranno un costo prossimo allo zero. Alcune tecniche di miglioramento delle reti neurali considerando le epoche sono la BackwardPropagation e la ForwardPropagation. [26, 28]



### 2.2.2 Vantaggi e difficoltà dell'utilizzo del Deep Learning

Una rete di deep learning offre diversi vantaggi rispetto al machine learning tradizionale, ma poiché è una tecnologia relativamente nuova, vi sono alcune difficoltà che derivano dalla sua implementazione pratica. [19]

*Vantaggi: [19].*

- **Elaborazione efficiente dei dati non strutturati:** I metodi di machine learning trovano i dati non strutturati, come i documenti di testo, difficili da elaborare perché il set di dati di addestramento può avere infinite variazioni. D'altra parte, i modelli di deep learning possono comprendere dati non strutturati e fare osservazioni generali senza l'estrazione manuale delle funzionalità. Ad esempio, una rete neurale può riconoscere che queste due diverse frasi di input hanno lo stesso significato:

Puoi dirmi come effettuare il pagamento?

Come trasferisco il denaro?

- **Relazioni nascoste e individuazione di modelli:** Un'applicazione di deep learning può analizzare grandi quantità di dati in modo più approfondito e rivelare nuove informazioni per le quali potrebbe non essere stata addestrata. Ad esempio, considerando un modello di deep learning addestrato per analizzare gli acquisti dei consumatori. Il modello ha dati solo per gli articoli che sono già stati acquistati. Tuttavia, la rete neurale artificiale può suggerire nuovi articoli che non hai acquistato confrontando i tuoi modelli di acquisto con quelli di altri clienti simili.

- **Elaborazione di dati instabili:** I set di dati instabili presentano grandi variazioni. Un esempio sono gli importi del rimborso di un prestito in banca. Una rete neurale di deep learning può classificare e ordinare anche quei dati, ad esempio analizzando le transazioni finanziarie e segnalandone alcune per il rilevamento delle frodi.

*Difficoltà: [19].*

- Grandi quantità di dati di alta qualità: Gli algoritmi di deep learning forniscono risultati migliori quando vengono addestrati su grandi quantità di dati di alta qualità. I valori anomali o gli errori nel set di dati di input possono influire in modo significativo sul processo di deep learning. Ad esempio, nel nostro esempio di immagini di animali, il modello di deep learning potrebbe classificare un aereo come tartaruga se immagini non di animali venissero accidentalmente introdotte nel set di dati. Per evitare tali imprecisioni, prima di poter addestrare modelli di deep learning è necessario pulire ed elaborare grandi quantità di dati. La pre-elaborazione dei dati di input richiede grandi quantità di capacità di archiviazione di dati.

- Grande potenza di elaborazione: Gli algoritmi di deep learning richiedono una elaborazione intensiva e un'infrastruttura con capacità di calcolo sufficiente per funzionare correttamente altrimenti impiegheranno molto tempo per elaborare i risultati.

## 2.3 Modelli di linguaggio per la generazione del testo

Un modello di linguaggio è un modello statistico che modella la distribuzione delle sequenze di parole, più in generale di sequenze di simboli discreti (lettere, fonemi, parole), in un linguaggio naturale. Un modello linguistico può, ad esempio, prevedere la parola che segue una sequenza di parole. [29]

I modelli di linguaggio possono essere usati per task come ad esempio:

- Machine Translation: [30] è utile assegnare una probabilità a una frase per avere traduzioni più corrette.
- Spell Correction: [31] è possibile usare una probabilità assegnata a una frase per correggere eventuali errori di ortografia.
- Text Generation: [18] è utile calcolare la probabilità della prossima parola in una sequenza, per ottenere la probabilità di una sequenza di parole e così generare una frase di senso compiuto.

Alcuni modelli di linguaggio per la generazione del testo sono:

- Modelli di linguaggio basati su n-grams. [32]
- Modelli di linguaggio basati su Deep Learning. [33]

I modelli di linguaggio più efficienti sono quelli basati su Deep Learning, poiché più accurati, grazie all'utilizzo di reti neurali. I modelli di linguaggio basati su Deep Learning possono essere utilizzati per catturare efficacemente le dipendenze e le proprietà sequenziali di una sequenza di input. [33]

Più precisamente, essi sono adatti per la modellazione e la generazione del testo, poiché sono in grado di soddisfare i seguenti aspetti: [33]

- generazione automatica di caratteristiche.
- cattura delle dipendenze a lungo termine e delle proprietà sequenziali.
- apprendimento end-to-end.
- Generalizzabilità.

Mentre, nei modelli a n-grams, anche se possono estrarre automaticamente le caratteristiche dal testo, essi non possono catturare bene tutte le dipendenze a lungo termine a causa della complessità esponenziale del calcolo combinatorio. [32, 34]

### 2.3.1 Modello linguistico basato su n-grams

Un modello linguistico statistico semplice ed efficace è il modello di linguaggio basato su n-grams. Questo modello presuppone che ogni parola sia condizionatamente dipendente dalle  $n - 1$  parole precedenti, attraverso la seguente formula: [34]

$$P(w_n | w_1, w_2, \dots, w_{n-1})$$

Questa formula rappresenta la probabilità condizionata. Il funzionamento di questo modello linguistico parte dalla suddivisione del testo in token<sup>1</sup>. Se il token viene rappresentato con i singoli caratteri, si può immaginare come la complessità dell'algoritmo aumenti, in più, il carattere successivo potrebbe essere una ripetizione del

---

<sup>1</sup>Un token è un'unità minima di analisi come ad esempio un carattere oppure una parola.

precedente. Se il token viene suddiviso in parole, allora la formula risulta essere molto efficiente sulla successiva parola da predire. Un vantaggio diretto degli n-grams è che sono facili da generalizzare. Gli n-grams non sono in grado di modellare le dipendenze a lungo termine tra i token, poiché si applica la formula su tutti i token del testo, come anche gli altri modelli statistici. Una possibile soluzione sarebbe troncare le informazioni posizionali a lungo termine, ovvero applicare la formula, ad esempio, solo agli ultimi 6-7 token inseriti nel testo (per ogni token). Inoltre, un'altra limitazione di un modello a n-grams è la scarsità della rappresentazione vettoriale della parola. Questo problema di scarsità può essere risolto utilizzando la rappresentazione distribuita dei modelli di linguaggio basati su Deep Learning. [34, 32]

### 2.3.2 Modelli linguistici basati su Deep Learning

I modelli linguistici basati su architetture neurali (neural networks) sono addestrate a predire parole mascherate in miliardi di frasi. Durante questo esercizio di predizione, le reti neurali regolano i propri parametri interni così da rappresentare nella propria struttura profonda, sia le relazioni tra le parole che il loro significato. Ed è proprio attraverso il linguaggio utilizzato che questi modelli linguistici imparano aspetti importanti della realtà [33]. Tali modelli possono usufruire di architetture di reti neurali di diverso tipo, come ad esempio: [33]

- FNN (Feedforward neural network): è una rete neurale artificiale dove le connessioni tra i nodi non formano cicli. Questo tipo di rete neurale fu la prima e la più semplice tra quelle messe a punto. In questa rete neurale le informazioni si muovono solo in una direzione, avanti, rispetto a nodi d'ingresso, attraverso nodi nascosti (se esistenti) fino ai nodi d'uscita. Nella rete non ci sono cicli. Le reti feed-forward non hanno memoria degli input avvenuti a tempi precedenti, per cui l'output è determinato solamente dall'attuale input. [35]
- RNN (Recurrent neural network): è una classe di rete neurale artificiale che include neuroni collegati tra loro in un ciclo. Tipicamente i valori di uscita di uno strato di un livello superiore sono utilizzati in ingresso di uno strato di livello inferiore. [36]

- LSTM (Long short-term memory): è una rete neurale artificiale utilizzata nei campi dell'intelligenza artificiale e del deep learning. A differenza delle reti neurali feedforward standard, LSTM ha connessioni di feedback. [37]
- Transformer: è un modello di apprendimento profondo che adotta il meccanismo dell'auto-attenzione, ponderando in modo differenziale il significato di ciascuna parte dei dati di input. [38]

Un modello di linguaggio basato su Deep Learning è la serie GPT di OpenAI. [29]

### 2.3.2.1 FNN

Una rete neurale feedforward è un tipo di rete neurale artificiale in cui le connessioni dei nodi non formano un ciclo. Spesso sono definite come una rete multistrato di neuroni. Le reti neurali feedforward sono così chiamate perché tutte le informazioni fluiscono solo in avanti. I dati entrano nei nodi in input, attraversano gli hidden layer ed infine escono dai nodi di output. La rete è priva di collegamenti che permettano alle informazioni che escono dal nodo di output di essere inviate nuovamente alla rete. Lo scopo delle reti neurali feedforward è quello di approssimare funzioni. Le prime architetture utilizzate per la generazione di testo furono le feedforward, queste architetture presentavano alcuni svantaggi come ad esempio:

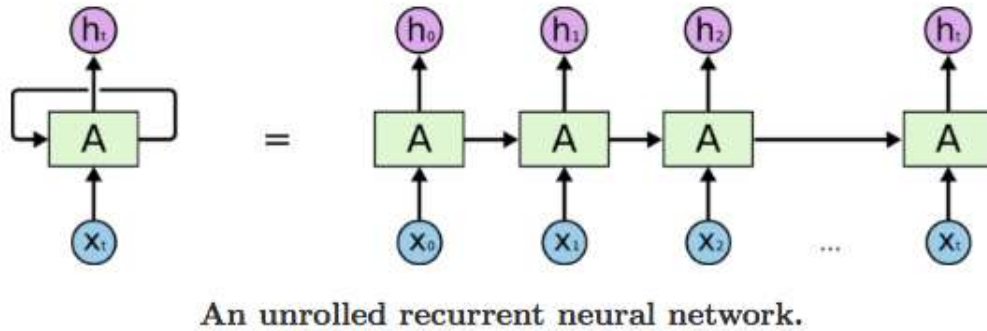
La taglia dell'input e dell'output: esse sono definite a priori prima di utilizzare la rete neurale, per cui, va definita prima la struttura statica. Immaginiamo di risolvere un problema in cui gli input sono i pixel di un'immagine, sapendo che non tutte le immagini hanno lo stesso numero di input. Ipotizzando di definire staticamente un input pari a 1080, ovvero 1080 pixel e considerando un'immagine da 480 pixel, allora quali valori verranno assegnati ai restanti pixel? Oppure, immaginiamo di avere un testo, esso può avere un numero variabile di caratteri, per cui, non possiamo definire un numero di input statico nella rete neurale.

Infine, per ottenere un miglioramento di prestazioni dei modelli di linguaggio, basati su deep learning, si può fare riferimento su architetture come RNN, LSTN e Transformer. [35, 39]

### 2.3.2.2 RNN

Recurrent Neural Network (RNN) è un tipo di rete neurale in cui l'output dello step precedente viene inviato come input allo step corrente. Nelle reti neurali tradizionali, tutti gli input e gli output sono indipendenti l'uno dall'altro, ma in casi come quando è necessario prevedere la parola successiva di una frase, sono richieste le parole precedenti e quindi è necessario ricordarle. Così è stata sviluppata la rete RNN, che ha risolto questo problema con l'aiuto di uno hidden layer. La caratteristica principale e più importante di RNN è lo stato nascosto, che ricorda alcune informazioni su una sequenza. RNN ha una "memoria" che ricorda tutte le informazioni su quanto è stato calcolato. Utilizza gli stessi parametri per ogni input poiché esegue la stessa attività su tutti gli input o hidden layer per produrre l'output. Ciò riduce la complessità dei parametri, a differenza di altre reti neurali. L'idea principale delle RNN è quella di elaborare in modo efficiente i dati sequenziali, poiché è fondamentale per prevedere i risultati in modo più preciso, soprattutto in ambito di NLP. Attraverso le RNN si risolve il problema della size dell'input e dell'output, infatti, permette di definirla in maniera ottimale. In genere, una rete neurale tradizionale elabora l'input e passa al successivo senza considerare alcuna sequenza. I dati sequenziali, invece, vengono elaborati seguendo un ordine specifico, necessario per comprenderli in modo distinto. Inoltre, sapendo che le RNN sono ottime per essere utilizzate per problemi che riguardano lunghe sequenze di input, si utilizza il metodo del gradiente (ovvero la derivata parziale, che permette di calibrare i pesi e il bias per il miglioramento della rete neurale). La RNN assegna lo stesso peso e lo stesso bias a ciascuno degli strati della rete. Pertanto, tutte le variabili indipendenti vengono convertite in variabili dipendenti. I loop della RNN garantiscono la conservazione delle informazioni nella sua memoria. Questo meccanismo viene denominato backpropagation ed è stato una grande aggiunta alla procedura di addestramento. L'obiettivo dell'utilizzo della backpropagation è quello di ripercorrere la rete neurale in modo da identificare qualsiasi derivata parziale dell'errore rispetto ai pesi. Un problema delle RNN è che sono difficili da addestrare, a causa del problema del vanishing gradient. Il problema del vanishing gradient avviene quando la norma del gradiente è molto piccola e l'aggiornamento dei pesi è inefficace, rendendo impossibile per il modello

apprendere la correlazione tra eventi temporalmente lontani, quindi non contribuisce al miglioramento dell'apprendimento della rete neurale. Inoltre, non sono in grado di conservare le informazioni passate su diverse scale temporali. Una specializzazione di questa architettura sono le LSTM che tentano di risolvere questo problema. [36, 40]

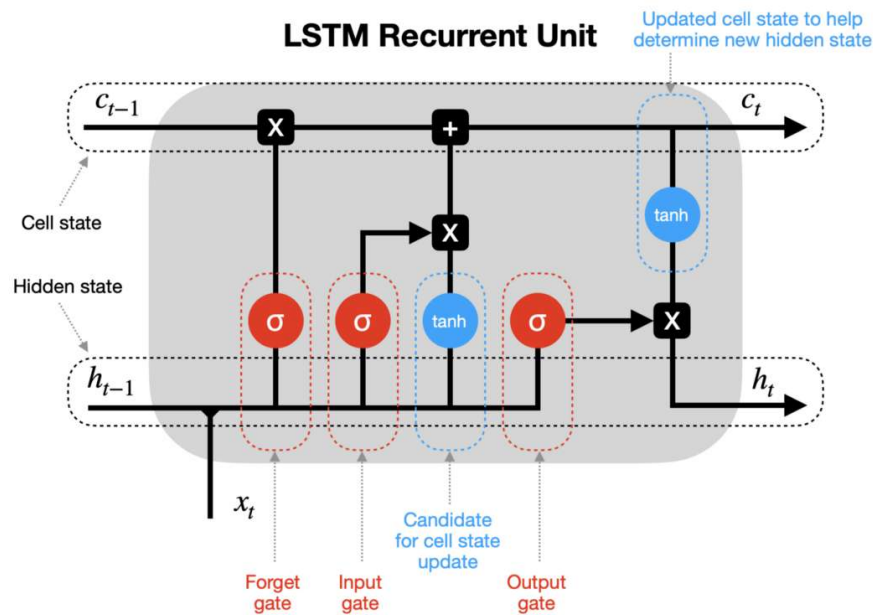


**Figura 2.7:** Recurrent Neural Network (RNN). [8]

### 2.3.2.3 LSTM

Una rete Long Short-Term Memory (LSTM) è una versione aggiornata della rete Recurrent Neural Network (RNN), per superare il problema del vanishing gradient. Una rete LSTM dispone di tre porte per controllare rispettivamente l'ingresso, l'uscita e la dimenticanza. Inoltre, c'è una cella di memoria che aiuta a trasportare le informazioni da una particolare istanza temporale all'istanza temporale successiva in modo efficiente. Quindi, può ricordare molte informazioni dagli stati precedenti rispetto alle reti RNN e superare il problema del vanishing gradient. Le reti LSTM sono comunemente usate nelle attività di NLP perché possono imparare il contesto richiesto per processare sequenze di dati. Le reti LSTM sono alimentate dai dati di input dall'istanza temporale corrente e dall'output del hidden layer dall'istanza temporale precedente.

# LONG SHORT-TERM MEMORY NEURAL NETWORKS



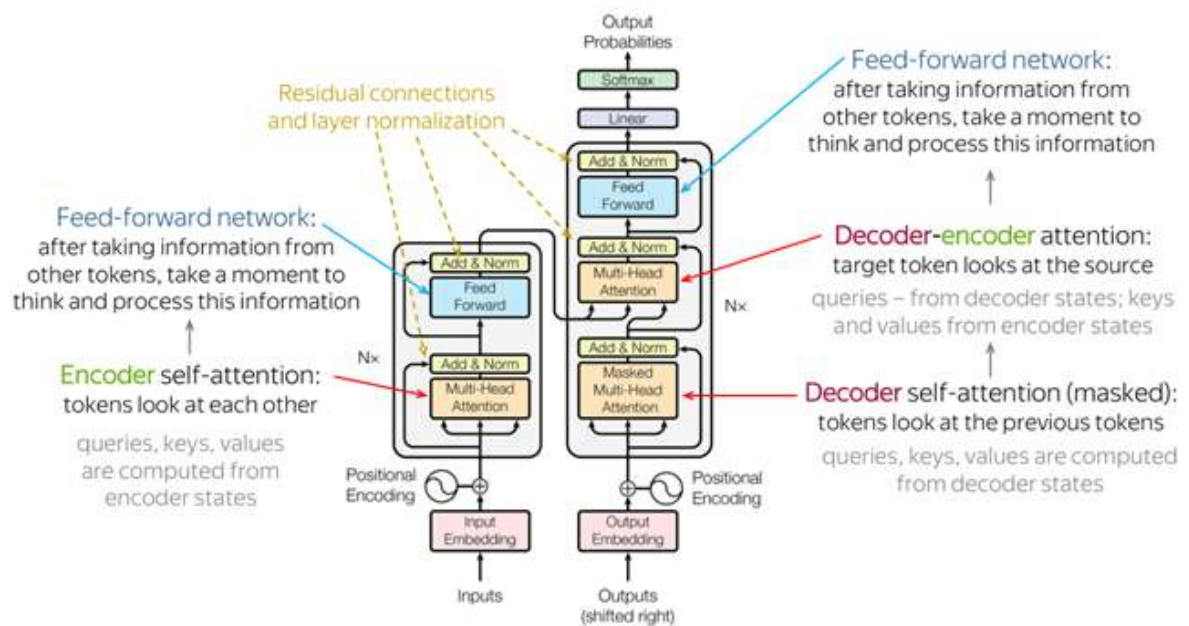
**Figura 2.8:** Long Short-Term Memory (LSTM). [9]

Possiamo notare che le reti LSTM sono molto simili alle reti RNN. Una differenza sostanziale con le reti RNN, è che ad ogni collegamento, vengono definiti dei gate (porte) che servono a definire dei calcoli precedenti, prima di far entrare un input all'interno di un neurone, per poi procedere con la funzione di attivazione. Questi calcoli, si occupano di determinare ciò che deve ricordare la rete neurale. Quindi si determina in quell'epoca, tutte quelle informazioni passate che la rete deve ricordare e quelle da dimenticare (informazioni meno rilevanti). Nell'NLG, le architetture LSTM, hanno una complessità computazionale maggiore, dato che una rete del genere risulta essere molto difficile da addestrare e parallelizzare, per cui subentrano le cosiddette architetture "Transformer". [37, 41]



### 2.3.2.4 Transformer

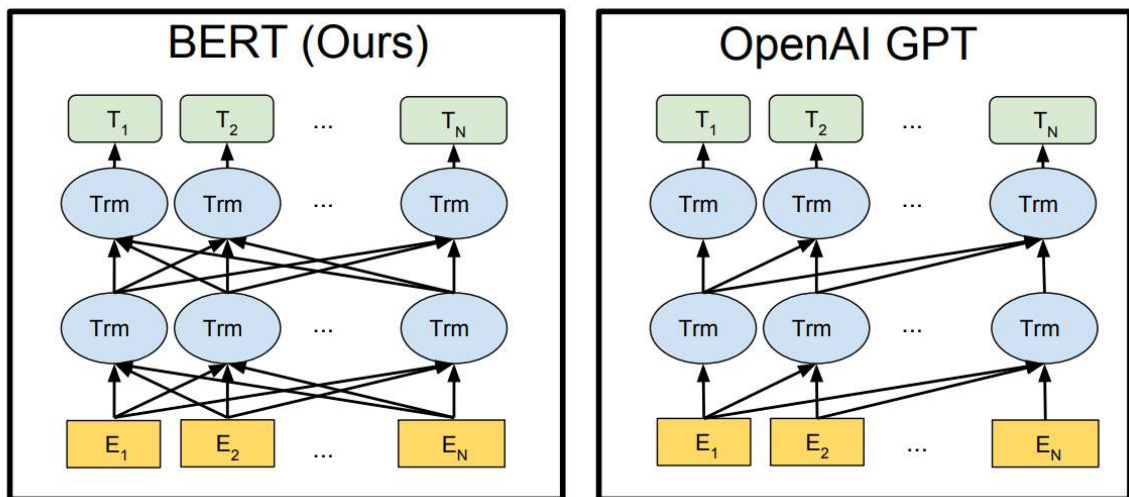
I transformer sono un particolare tipo di rete neurale che apprendono il contesto e quindi il significato tracciando le relazioni in dati sequenziali. I modelli Transformer applicano una serie di tecniche matematiche in evoluzione, chiamate attention o self-attention, per individuare i modi in cui elementi di dati in una serie, anche distanti tra loro, si influenzano e dipendono l'uno dall'altro. Notiamo nella Figura 2.9 come una



**Figura 2.9:** Rappresentazione di una rete Transformer. [10]

architettura di rete neurale transformer è composta da due componenti principali: encoder e decoder. L'encoder mappa la sequenza di simboli in input ( $x_1, \dots, x_n$ ) in una sequenza di rappresentazioni continue  $z = (z_1, \dots, z_n)$ . Data la sequenza  $z$ , il decoder genera una sequenza di output ( $y_1, \dots, y_m$ ) di simboli, generando un elemento alla volta. Ad ogni passo il modello è auto-regressivo ed utilizza i simboli generati precedentemente come un input aggiuntivo per produrre il prossimo. Come le RNN, i transformer sono utili per processare dati di input sequenziali, solo che a differenza delle reti RNN, processano i dati in una volta sola. Questo avviene attraverso la self-attention, che permette al trasformatore di non elaborare una parola alla volta. Ciò consente una maggiore parallelizzazione rispetto alle RNN e quindi riduce i tempi di addestramento. I transformer sono emersi grazie ai miglioramenti che hanno apportato in termini di efficienza e accuratezza nella gestione di task riguardanti il

Natural Language Processing. Queste reti non solo apprendono ripetendo le stesse azioni ma trovano anche pattern nei dati, imparando dal contesto con lo scopo di creare nuove informazioni. I transformer hanno cambiato considerevolmente il modo con cui lavoriamo con i dati testuali, tuttavia presentano alcune limitazioni. In particolare, l'attention può gestire solo stringhe di lunghezza fissata. Il testo viene quindi suddiviso in segmenti prima di essere inviato al sistema come input. È proprio questo accorciamento del testo che può portare alla frammentazione contestuale, con conseguente scissione di parti importanti del testo. Questo tipo di rete neurale ha portato alla realizzazione di modelli come: GPT, BERT... [38, 42]

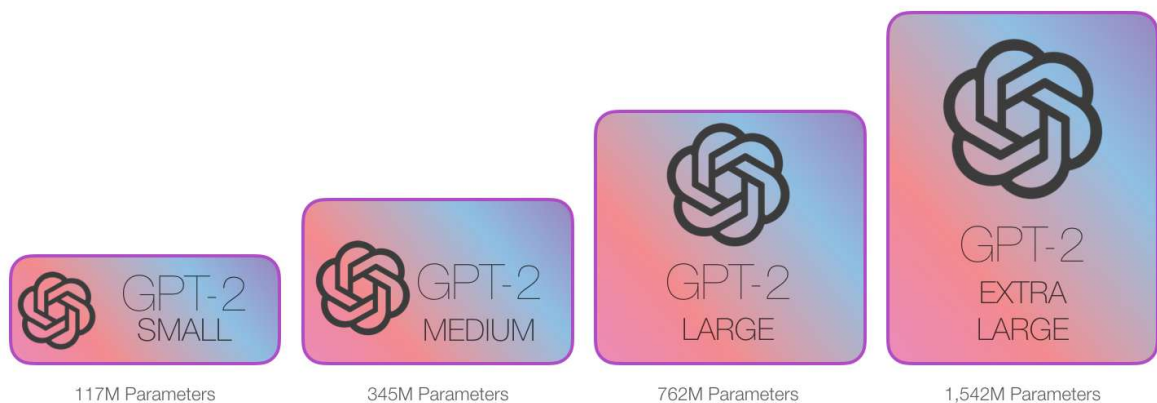


**Figura 2.10:** Modelli GPT, BERT. [11]

## 2.4 GPT-2

GPT-2 è l'acronimo di Generative Pre-trained Transformer 2, ed è un'intelligenza artificiale open source creata da OpenAI, che si occupa di generare testo a partire da input. Essa risulta essere molto utile anche per generare summary, tradurre del testo e rispondere a delle domande. GPT-2 è la seconda generazione della serie GPT. GPT è uno dei modelli più importanti e fondamentali per la comprensione del linguaggio, che ha contribuito a gettare le basi della modellazione linguistica. Questo modello è anche uno dei pionieri della nascita della NLP, per quanto riguarda l'elevato numero di parametri di addestramento, con 110 milioni di parametri (che ad oggi possono

sembrare pochi, ma all'epoca della sua uscita erano davvero tanti), ottenuto grazie all'utilizzo di una delle prime architetture Transformer. GPT si basa essenzialmente sul concetto di pre-addestramento e di transfer learning di un modello linguistico su un enorme corpus di dati e di successiva messa a punto. GPT-2 risulta essere più performante rispetto al suo predecessore, infatti, risulta essere 10 volte migliore di GPT, poiché offre 1,5 miliardi di parametri di addestramento ed inoltre è pre-addestrato su un dataset di 40GB. Esistono diverse varianti di GPT-2, ognuna suddivisa per lo spazio di memoria che occupa per memorizzare il modello sulla base dei suoi parametri. La variante più piccola di GPT-2 addestrato, occupa 500 MB di spazio di archiviazione per memorizzare tutti i suoi parametri. Mentre l'ultima variante di GPT-2 è 13 volte più grande, quindi potrebbe occupare più di 6,5 GB di spazio di archiviazione. [43, 12]

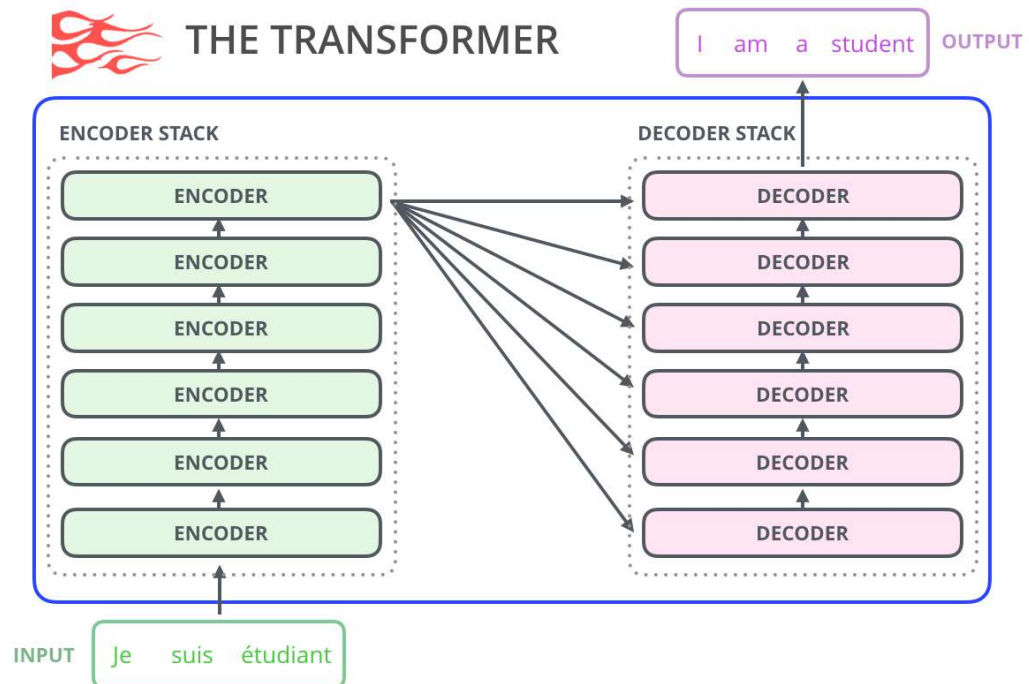


**Figura 2.11:** Varianti del modello di GPT-2. [12]

Per ottenere un miglioramento delle prestazioni, OpenAI ha sviluppato una nuova generazione di GPT, chiamata GPT-3. GPT-3 utilizza la stessa tipologia di architettura di GPT-2, ma la sostanziale differenza è che offre 175miliardi di parametri di addestramento contro i 1,5miliardi di GPT-2, ed è pre-addestrato su 45TB di testo rispetto ai 40GB del suo predecessore, inoltre, un altro suo vantaggio è la possibilità di poter effettuare il Few-shot learning. Infine, GPT-3 è closed source. [44]

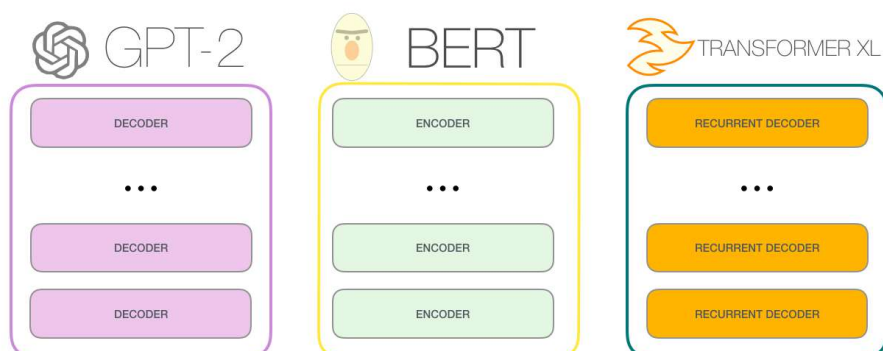
### 2.4.1 Architettura e funzionamento

[12] GPT-2 si basa su un'architettura di rete neurale di tipo Transformer. Nell'architettura dei Transformer classica, essa è composta da encoder e decoder, infatti:



**Figura 2.12:** Architettura transformer. [12]

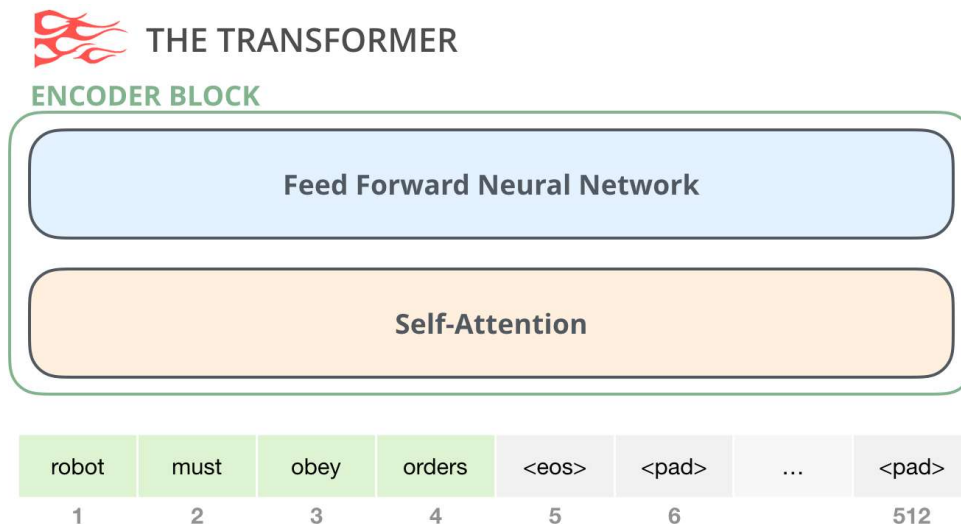
Gran parte del lavoro di ricerca successivo, ha visto l'architettura liberarsi di uno dei due blocchi, encoder decoder, ottenendo:



**Figura 2.13:** Modelli GPT, BERT, TRANSFORMER XL. [12]

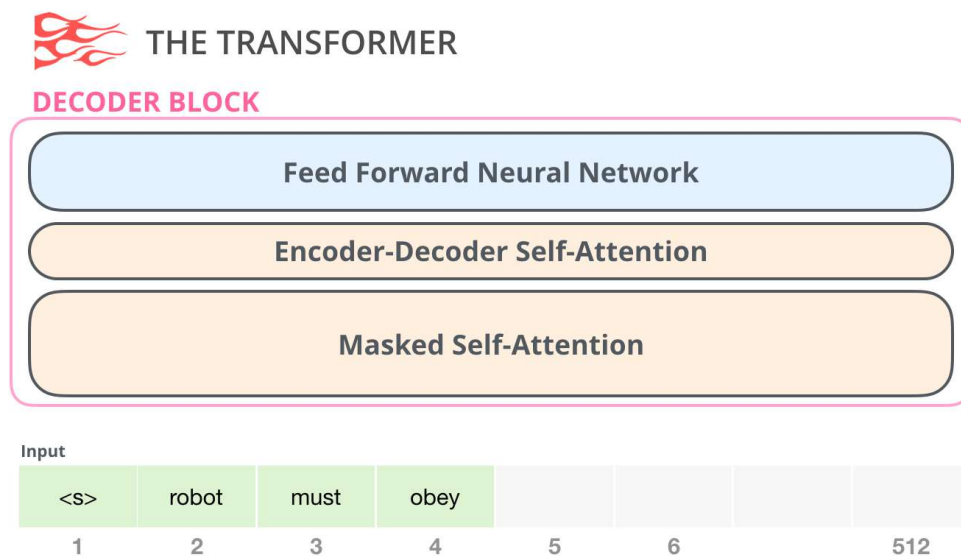
Possiamo notare che una differenza architetturale tra GPT-2 e BERT sono i blocchi di encoder e decoder, infatti GPT-2 usa solo blocchi di decoder, mentre BERT solo blocchi di encoder. Inoltre, il blocco dei decoder e degli encoder di BERT e GPT-2 sono diversi dai blocchi classici, poiché, si è avuto un'evoluzione dei blocchi delle architetture Transformer, che risultano essere:

- Per gli encoder:



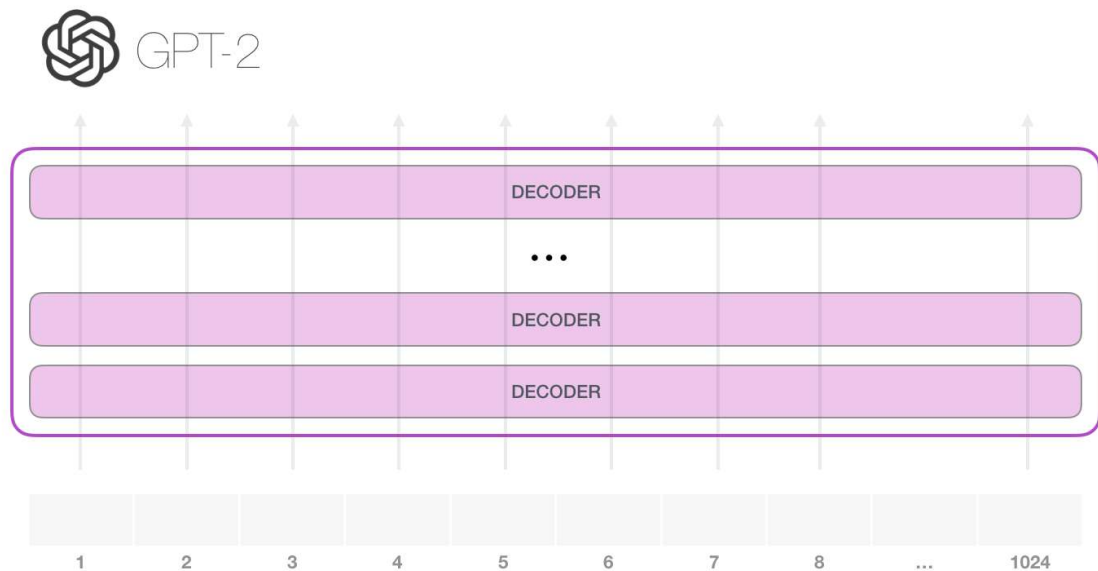
**Figura 2.14:** Blocco Encoder. [12]

- Per i decoder:



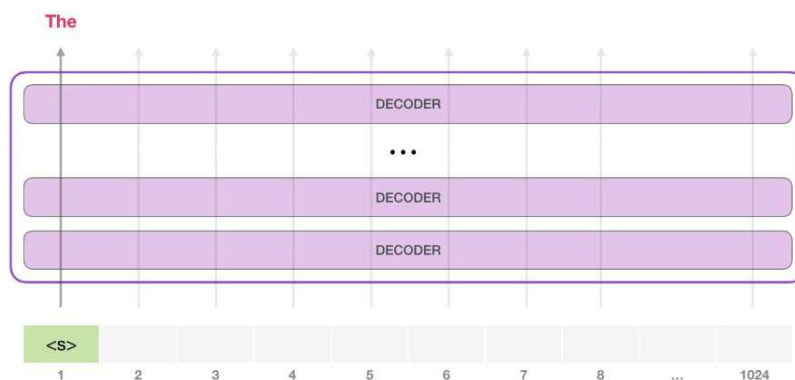
**Figura 2.15:** Blocco Decoder. [12]

Una differenza sostanziale è che GPT-2 costruisce la frase un token alla volta. Dopo che un token è stato generato, otteniamo una frase parziale, che sarà considerata come un nuovo input per il modello nella fase successiva. Il nuovo input sarà processato per generare un nuovo token, che verrà aggiunto come ultima parola, in modo da ripetere il processo, ottenendo una frase finale. Quest'idea prende il nome di auto-regressione. Ad esempio, GPT-2 genera la parola "PIANO" dopo un input, la parola "PIANO" sarà considerata come nuovo input. GPT-2 genererà un'altra parola come ad esempio "FORTE", concatenando "FORTE" e "PIANO", si ottiene "PIANOFORTE", che sarà un nuovo input del modello, in modo da generare il prossimo token e di concatenarlo alla frase di input, infine si ripete il processo.

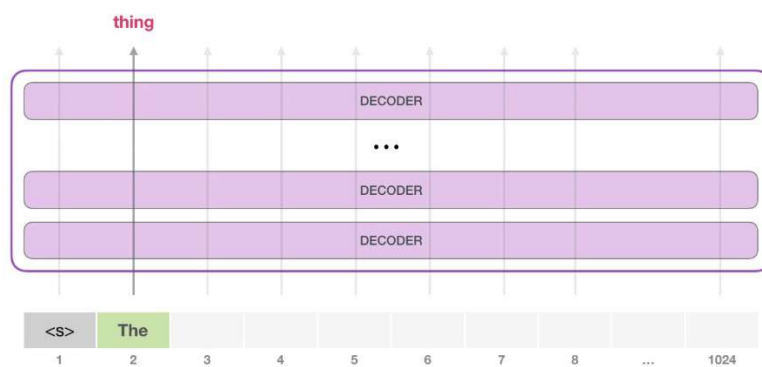


**Figura 2.16:** Generazione token(1). [12]

Come si può notare dalla Figura 2.16, il numero massimo di token che GPT-2 può utilizzare è 1024. Per la predizione di un token si utilizza la masked self-attention, ovvero per generare un token ci si basa su tutti i token precedenti ad esso. Più in dettaglio per la predizione di un nuovo token, tutti i token precedenti sono stati dati in input a tutti i decoder.



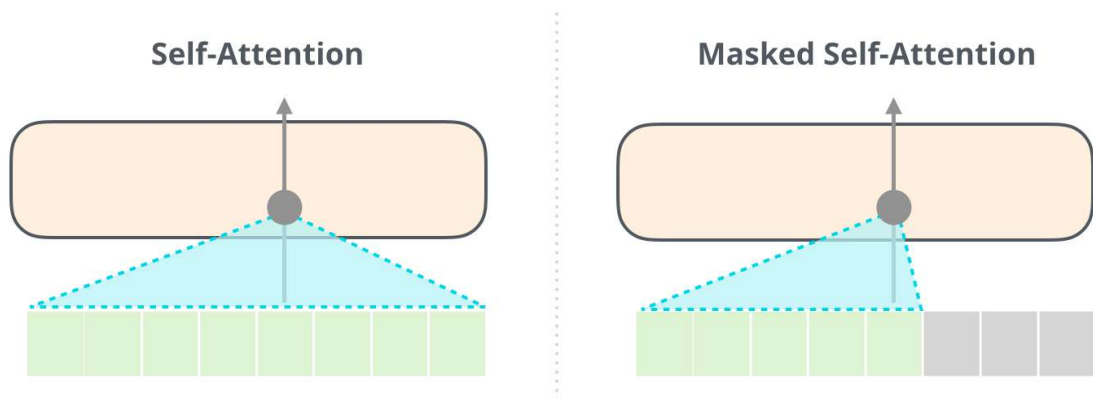
**Figura 2.17:** Generazione token(2). [12]



**Figura 2.18:** Generazione token(3). [12]

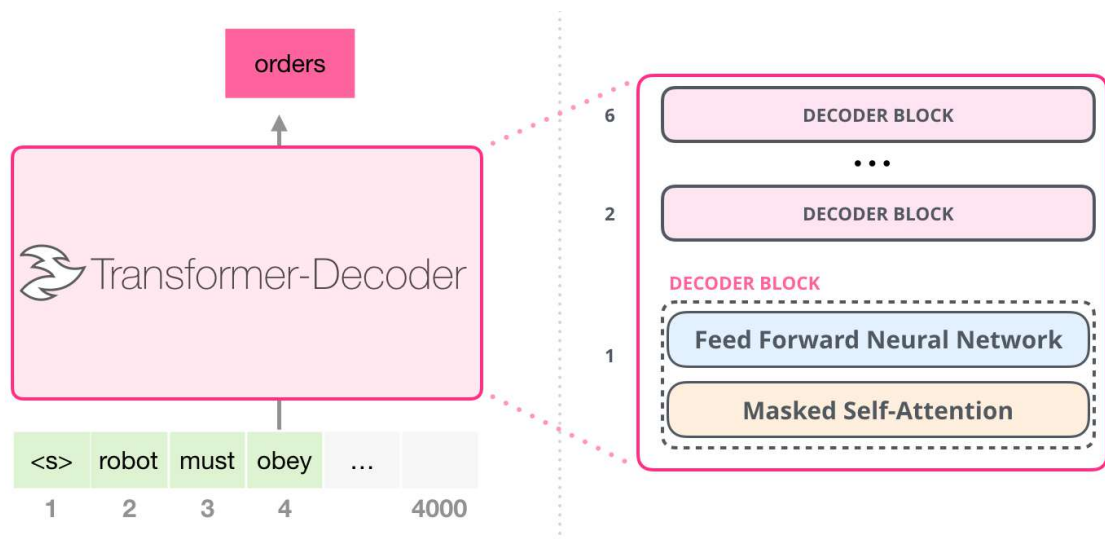
La masked self-attention del blocco dei Decoder si differisce dalla Self-Attention utilizzata dal blocco degli Encoder. Infatti, supponiamo di avere massimo 1024 token sia su GPT-2 che su BERT, e supponiamo di aver riempito solo i primi 4 token, ovvero quei token che mi rappresentano la frase parziale nella quinta iterata. GPT-2, attraverso l'utilizzo della masked self-attention, va a mascherare tutti i token, in modo da permettere la generazione della nuova parola nella quinta iterazione applicando la self-attention, ovvero il calcolo della previsione per la generazione della nuova parola, tenendo conto solo dei primi 4 token. BERT invece, andrà ad applicare la self-attention su tutti i token. Questo però potrebbe influenzare negativamente il calcolo della previsione della generazione della nuova parola.





**Figura 2.19:** Differenze tra Self-Attention e Masked Self-Attention. [12]

Il blocco decoder, in generale, dispone di uno strato successivo alla masked self-attention, che prende il nome di encoder-decoder-self-attention. Questo strato si basa su una self-attention ottenuta sia dall'encoder che dal decoder. Esso viene utilizzato solo nell'architetture dei Transformer, che hanno sia blocchi di encoder che di decoder. Nel caso di GPT-2 non avendo Encoder, questo strato viene eliminato. Infine, per ogni blocco encoder | decoder si ha una rete neurale del tipo FFNN.



**Figura 2.20:** Architettura GPT-2. [12]



[13] GPT-2-Simple è una libreria custom che racchiude gli script esistenti per la messa a punto e la generazione del modello GPT-2 di OpenAI (in particolare le versioni "piccola" da 124M e "media" da 355M iperparametri). Inoltre, questo pacchetto consente una generazione più semplice del testo, la generazione in un file per una facile gestione, consentendo l'uso di prefissi per forzare il testo a iniziare con una determinata frase. Questo pacchetto incorpora e apporta modifiche minime di basso livello a:

- Gestione del modello dal repo ufficiale GPT-2 di OpenAI (licenza MIT)
- Finetuning del modello dal fork di Neil Shepperd di GPT-2 (licenza MIT)
- Gestione dell'output di generazione del testo da textgenrnn (Licenza MIT)

Per la messa a punto è fortemente consigliato l'uso di una GPU, anche se è possibile generare usando una CPU (il processo è più lento). Un'altra libreria di supporto per GPT-2-simple è TensorFlow. TensorFlow è una libreria software open source per l'apprendimento automatico, che fornisce moduli sperimentali e ottimizzati, utili nella realizzazione di algoritmi per diversi tipi di compiti percettivi e di comprensione del linguaggio. Questa libreria svolge diverse funzioni di supporto come ad esempio:

- Il salvataggio del modello in un determinato checkpoint.
- Il caricamento di un modello in un determinato checkpoint.
- possibilità di utilizzare la potenza computazionale della gpu per un aumento di prestazioni.

Lo sviluppo di gpt-2-simple è stato in gran parte superato da aitextgen, che ha capacità simili di generazione di testo AI con tempi di addestramento e utilizzo delle risorse più efficienti. Se non si desidera utilizzare TensorFlow, è consigliabile utilizzare aitextgen. I checkpoint addestrati con gpt-2-simple possono essere caricati anche con aitextgen.

### 2.4.1.1 Utilizzo di GPT-2-simple

[13] Per importare la libreria GPT-2-simple in python si utilizza il seguente comando:

```
1 import gpt_2_simple as gpt2
```

Per eseguire il download del modello 774M di GPT-2 si utilizza:

```
1 gpt2.download_gpt2("774M")
```

il modello viene salvato nella directory corrente con una specifica struttura.

Le funzioni di GPT-2-simple hanno bisogno di una sessione TensorFlow, che la si ottiene nel seguente modo:

```
1 sess = gpt2.start_tf_sess()
```

Il comando per eseguire il finetuning è il seguente:

```
1 gpt2.finetune(sess, 'dataset.txt', steps=1000)
2 # steps is max number of training steps
```

- sess: sessione tensorflow;
- dataset.txt: risulta essere il file di addestramento.
- steps: indica il numero di epoche di training da effettuare. Un'epoca di training si realizza quando il modello viene allenato su tutte le istanze del dataset una volta.

Inoltre, ci sono anche altri parametri come:

- Model\_name: è il nome del modello utilizzato.
- Restore\_from: può avere due valori, "fresh" e "latest" e indica, nel primo caso, se il modello debba ricominciare da zero il training oppure, nel secondo caso, se deve ripartire dall'ultimo checkpoint memorizzato.
- Run\_name: indica il path dell'ultimo checkpoint salvato.
- Print\_every: indica il numero di steps dopo il quale stampa i progressi fatti dal

modello riguardo al training. In particolare, stampa i valori di loss e loss avg che indicano, rispettivamente, i valori correnti e la media dei valori di loss.

- `Sample_every`: indica il numero di steps dopo il quale stampa una frase di prova generata senza nessun input.
- `Save_every`: indica il numero di steps dopo il quale il modello memorizza un checkpoint.

Una volta eseguito il finetuning del modello, esso potrà generare del testo. Per la generazione del testo si userà un comando del tipo:

```
1 single_text = gpt2.generate(sess, return_as_list=True) [0]  
2 print(single_text)
```

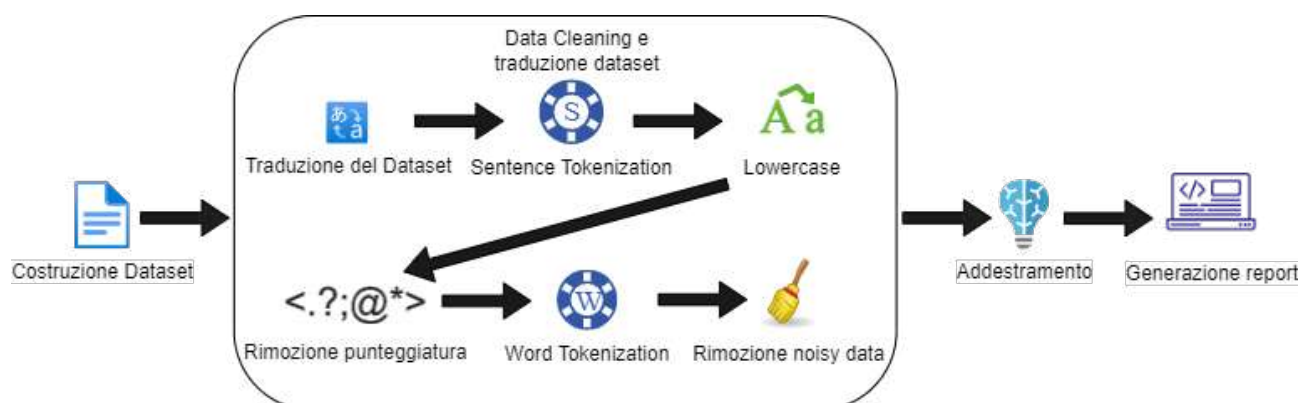
Per ottenere un output più sistematico, si possono utilizzare diversi parametri come:

- `prefix`: indica il prefisso con cui dovrebbe iniziare la parola.
- `length`: indica la lunghezza dei token. GPT-2 ne mette a disposizione al più 1024.
- `temperature`: indica il grado di “fantasia” con cui il modello esegue gli accoppiamenti di parole per generare dei periodi originali. Questo valore è un float, ovvero un decimale, che può andare da 0.1 a 1, e i suoi valori consigliati sono: 0.5 per 60 parole, tra 0.6 e 0.8 per minimo 100 parole, e in generale valori maggiori di 0.6 per più di 100 parole con un training set molto grande.
- `nsamples`: indica il numero di esempi di frasi generate con lo stesso input.

## CAPITOLO 3

### Implementazione

In questo capitolo ci si addenterà nello sviluppo del sistema di intelligenza artificiale per la generazione di report ambientali inerenti al territorio nazionale. Il primo passo da eseguire sarà quello di costruire un dataset per il modello, estraendo testo da file PDF contenenti report sulla qualità dell'aria. Il secondo passo da svolgere sarà quello di effettuare una "pulizia" dei dati, cercando di rimuovere i dati "rumorosi" all'interno del nostro dataset, attraverso un processo di data cleaning. Subito dopo, si prosegue con l'addestramento attraverso un modello basato su Transformers, in particolare si vedrà l'utilizzo di GPT-2, che si è mostrato nella sezione 2.4. Infine, si passerà alla generazione automatica del report ambientale.



**Figura 3.1:** Pipeline implementazione.

## 3.1 Specifiche tecniche e librerie utilizzate

Le librerie utilizzate per lo sviluppo del modello sono molteplici:

- GPT-2-SIMPLE: utilizzata per la costruzione del modello di GPT2. Questa libreria ha il vantaggio di essere usata in maniera del tutto black box, infatti, ad esempio durante il fine-tuning, l'utilizzatore non dovrà nemmeno richiamare ulteriori metodi per l'uso del tokenizer. La libreria inoltre fornisce varie versioni di GPT2, che risultano essere: 124M, 355M, 774M e 1558M.

**Tabella 3.1:** Differenze delle versioni fornite da GPT-2-SIMPLE [13].

Versione	Peso	Fine-Tuning
124M	circa 500MB	SI
335M	circa 1.5GB	SI
774M	circa 3-4GB	SI
1558M	circa 6-7GB	NO

***Nota:** Per "Fine-Tuning" si indica se è possibile effettuare l'addestramento su una specifica versione, data la libreria in questione.*

Come si può vedere dalla tabella, sulla versione 1558M non è possibile effettuare il fine-tuning, ed ecco perché la versione utilizzata nel nostro sistema risulta essere la 774M. Per poter utilizzare la versione 774M, si è utilizzata una macchina dotata di una CPU per server e di 64GB di RAM. La libreria GPT-2-SIMPLE può essere utilizzata anche attraverso l'aiuto di GPU con supporto driver CUDA, il che migliora le prestazioni in termini di velocità durante l'addestramento e la generazione. Per poter usufruire di una GPU, c'è bisogno di un minimo di GB da supportare.

Il codice per usufruire di GPT-2-SIMPLE e della versione 774M risulta essere:

```
1 pip3 install gpt-2-simple
```

Per eseguire il download della versione 774M è stato utilizzato il seguente codice.

```
1 import gpt_2_simple as gpt2
2
3 gpt2.download_gpt2(model_name="774M")
```

- TensorFlow:è una libreria indispensabile che fa da supporto a GPT-2-SIMPLE, può essere utilizzata per decidere se integrare o meno l’uso della gpu e permette di creare sessioni.

```
1 pip3 install tensorflow.
```

- Googletrans:è una libreria che implementa l’API di Google Translate. Questa libreria è stata utilizzata per effettuare la traduzione del dataset in inglese.

```
1 pip3 install googletrans
```

- nltk: Natural Language Toolkit è una libreria utilizzata per l’elaborazione del linguaggio naturale. In particolare è stata utilizzata nltk.tokenize per effettuare una fase del processo di data cleaning, la word tokenization e sentence tokenization.

```
1 pip3 install nltk
```

## 3.2 Costruzione del dataset

Per la costruzione del dataset si è deciso di raccogliere i dati estraendoli da vari file PDF contenenti report sulla qualità dell'aria. Sono stati utilizzati due algoritmi creati ad hoc per il problema in esame che utilizzano il tool PDFPlumber in modo da permettere l'estrazione da file PDF. I due algoritmi sviluppati hanno come obbiettivo quello di estrarre testo da file PDF e al contempo cercare di eliminare, anche se in parte, testo che non risulta essere inerente al problema.

- Il primo algoritmo creato è il seguente:

```

1  from asyncore import loop, write
2  from turtle import clear
3  import pdfplumber
4  from array import *
5
6  f = open('Prova1.txt', 'a', encoding='UTF-8')
7  contatore = array('l', [])
8  lung = array('l', [])
9  with pdfplumber.open('Rapporto_Malaria_2021.pdf', laparams={}) as pdf:
10
11     print('pagina iniziale ')
12     a = input()
13     pagina_inizio = int(a)
14     print('pagina fine ')
15     b = input()
16     pagina_fine = int(b)
17
18     while(pagina_inizio<=pagina_fine and loop!=0):
19         loop=1
20         salto=1
21
22         while(pagina_inizio<=pagina_fine):
23             print('vuoi saltare la pagina=', pagina_inizio+1, '?1=SI, 0=NO')
24             jump = input()
25             salto = int(jump)
26             if(salto==1):
27                 print('pagina=', pagina_inizio, ' saltata')
28             elif(salto==0):
29                 contatore.append(pagina_inizio)
30                 if(pagina_inizio==pagina_fine):
31                     contatore.append(pagina_fine)
32                 pagina_inizio+=1

```

```

33     o=0
34     totpag=len(contatore)
35     print('numero di pagine=',totpag)
36     while(o<totpag):
37         m=contatore[o]
38         page = pdf.pages[m]
39         dizionario = page.extract_words(x_tolerance=3, y_tolerance=3,
40         keep_blank_chars=False, use_text_flow=False,
41         horizontal_ltr=True, vertical_ttb=True,
42         extra_attrs=[], split_at_punctuation=False)
43
44         numero=len(dizionario)
45         num=numero-1
46         lung.append(num)
47         o+=1
48
49         k=0
50
51         while(k<=num):
52             f.write(dizionario[k]['text'])
53             f.write(' ')
54             k+=1
55     f.close

```

Questo algoritmo, ad ogni pagina, chiede all'utente se il testo all'interno della pagina debba essere estratto o meno, in modo da poter inserire nel nostro dataset solo testo inerente alla qualità dell'aria.

•Secondo algoritmo creato:

```

1  from asyncio import loop, write
2  from turtle import clear
3  import pdfplumber
4
5  f = open('Prova.txt', 'a', encoding='UTF-8')
6
7  with pdfplumber.open('Rapporto_Malaria_2021.pdf', laparams={}) as pdf:
8
9      print('pagina iniziale ')
10     a = input()
11     pagina_inizio = int(a)
12     print('pagina fine ')
13     b = input()
14     pagina_fine = int(b)

```



```

15
16 while(pagina_inizio<=pagina_fine and loop!=0):
17     loop=1
18     salto=1
19     page = pdf.pages[pagina_inizio]
20     dizionario = page.extract_words(x_tolerance=3, y_tolerance=3,
21     keep_blank_chars=False, use_text_flow=False, horizontal_ltr=True,
22     vertical_ttb=True, extra_attrs=[], split_at_punctuation=False)
23
24     numero=len(dizionario)
25     #numero_tot+=numero
26     print('numeri di parole',numero)
27     print('\n')
28     print('numero pagina=',pagina_inizio+1)
29     print('\n')
30     print('da che parola vuoi iniziare?')
31     inizio = input()
32     i = int(inizio)
33     print('inizi da= ',dizionario[i]['text'])
34     k=i
35
36     print('a che parola vuoi finire?')
37     fine = input()
38     j = int(fine)
39     print('finisci a= ',dizionario[j]['text'])
40
41     while(i<j):
42         print(dizionario[i]['text'])
43         i+=1
44
45     print('va bene? 1=SI 0=NO')
46     scelta = input()
47     risposta = int(scelta)
48
49     if (risposta==1):
50         while(k<=j):
51             f.write(dizionario[k]['text'])
52             f.write(' ')
53             k+=1
54             print('preso')
55             print('La prima parola presa è',inizio)
56             print('Lultima parola presa è',fine)
57             pagina_inizio+=1
58     elif (risposta==0):
59         print('non preso')
60         print('Non hai preso da=',inizio, ' alla parola=',fine)
61     print('vuoi continuare? premi un numero per continuare oppure 0')

```

```

62     continuo = input()
63     loop = int(continuo)
64
65
66     while(salto==1 and pagina_inizio<=pagina_fine):
67         print('vuoi saltare la pagina=',pagina_inizio+1,'?1=SI,0=NO')
68         jump = input()
69         salto = int(jump)
70         if(salto==1):
71             pagina_inizio+=1
72             print('pagina=',pagina_inizio,' saltata')
73
74 f.close

```

Questo algoritmo, a differenza di quello visto precedentemente, risulta essere più dispendioso in termini di tempo poiché cerca di estrarre testo andando ad analizzare ogni carattere, dando in input il carattere iniziale da cui procedere all'estrazione e il carattere alla quale fermarsi. Questo processo viene svolto ad ogni pagina, in modo da iniziare ad eliminare dati non inerenti al problema.

Una volta svolto l'estrazione dei dati dai file PDF abbiamo ottenuto il nostro dataset avente una dimensione di circa 2 MB di testo. Questo dataset ovviamente è molto piccolo, ma è il massimo che siamo stati in grado di ottenere.

Presentiamo qui una piccola parte del dataset.

```

Firenze, 29 Gennaio 2016 Comunicato Stampa Polveri fini, biossido di azoto e ozono
migliorano complessivamente le città toscane ma ci sono ancora molte criticità Mal'Aria
di città 2016
I dati regionali sull'inquinamento atmosferico Il 2015 si è concluso all'insegna
dell'emergenza smog. La maggior parte delle città toscane si è "svegliata" ancora una
volta, verso la metà di dicembre,
con le centraline di fondo urbano e di traffico che registravano, quasi
ininterrottamente, per due settimane, sforamenti del PM10 sopra il valore limite di 50
microgrammi per mc.
Un limite che per legge non può essere superato più di 35 volte in un anno. Sono dati
emersi dalla conferenza stampa di presentazione del dossier di Legambiente Mal'aria 2016
che si è svolta stamane al caffè letterario Giubbe Rosse,
in piazza della Repubblica, alla presenza del presidente regionale di Legambiente Fausto
Ferruzza e della direttrice generale ARPAT Maria Sargentini.
"La situazione media è decisamente migliorata e non possiamo che esserne felici -
dichiara Fausto Ferruzza, Presidente di Legambiente Toscana - tuttavia proprio oggi che
commentiamo dati tutto sommato lusinghieri,
non possiamo dimenticare l'emergenza smog di dicembre. Tanti giorni consecutivi di alta
pressione, di nebbia e di assenza totale di vento hanno creato solo poche settimane fa un
cocktail micidiale da allarme sanitario.

```

**Figura 3.2:** Dataset ottenuto tramite l'estrazione.

## 3.3 Tecniche di Data Cleaning e traduzione del dataset

### 3.3.1 Traduzione del dataset

Dopo la costruzione del dataset si è passato alla traduzione di esso, tramite un apposito algoritmo. La traduzione è stata fatta da italiano ad inglese. Le motivazioni per la quale il dataset è stato tradotto sono principalmente due:

- La prima motivazione è legata al modello stesso, poiché si è riscontrato che Gpt-2 lavora meglio con delle frasi scritte in inglese. Infatti, possiamo notare come il modello Gpt-2 è pre-addestrato su dati in inglese.
- La seconda motivazione è legata al processo di Data Cleaning, infatti, eseguire questo processo su frasi in inglese risulta essere più efficiente.

```
1 from cgitb import text
2 import linecache
3 from googletrans import Translator
4
5 trans = Translator()
6
7 f = open("Prova.txt", mode='a', encoding='UTF-8')
8 c=1
9
10 while(True):
11     data = linecache.getline('b.txt',c).strip()
12     out = trans.translate(data,src="it",dest="en")
13     c+=1
14     print(out.text)
15     f.write(out.text)
16     f.write("\n")
17     if not data:
18         print("end of file")
19     break
```

Di seguito è mostrato il dataset tradotto in inglese.

Florence, 29 January 2016 Press Release Fine dust, nitrogen dioxide and ozone improve overall Tuscan cities but there are still many critical issues  
Mal'Aria di città 2016 Regional data on air pollution 2015 ended under the banner of the smog emergency. Most of the Tuscan cities "woke up" once again, around mid-December, with the urban background and traffic control units which recorded, almost continuously, for two weeks, exceedances of the PM10 above the limit value of 50 micrograms per cubic meter. A limit that by law cannot be exceeded more than 35 times in a year. These data emerged from the press conference for the presentation of the Legambiente Mal'aria 2016 dossier which took place this morning at the Giubbe Rosse literary café, in Piazza della Repubblica, in the presence of the regional president of Legambiente Fausto Ferruzza and the general director of ARPAT Maria Sargentini. "The average situation has definitely improved and we can only be happy about it - declares Fausto Ferruzza, President of Legambiente Toscana - however, today that we are commenting on all in all flattering data, we cannot forget the smog emergency in December. Many consecutive days of high pressure, fog and total absence of wind created a deadly cocktail of health alarms just a few weeks ago. The plan to prevent malaria in the city is known: care of the iron in local public transport, more cycle networks, more pedestrianizations in our ancient centres; domestic heating with the highest rate of innovation in industrial activities rigorous application of the European principle of the polluter pays, in view of a progressive reduction in the emission intensity of our production activities". The smog emergency during the winter

**Figura 3.3:** Dataset tradotto.

### 3.3.2 Sentence tokenization

Una volta aver effettuato la traduzione del dataset si è passato al processo di Data Cleaning, più in particolare alla Sentence Tokenizzazione. Come visto in precedenza, “Tokenizzare” un testo significa suddividerlo in unità minime di analisi (chiamate token). L’unità minima di analisi scelta è la frase (Sentence), questa tecnica è stata utile per dare una prima forma al nostro dataset in modo tale che il nostro modello iniziasse ad addestrarsi non più su singole stringhe ma su frasi di senso compiuto.

L’algoritmo utilizzato è il seguente:

```
1  from asyncore import write
2  from cgitb import text
3  import nltk
4  from nltk.tokenize import sent_tokenize
5  import linecache
6
7  nltk.download('punkt')
8  c=1
9  f = open("Prova.txt", mode='a', encoding='UTF-8')
10 while (True):
11     data = linecache.getline('datasetIng.txt', c)
12     sentence_tokenizer_output = sent_tokenize(data)
13     c+=1
14     for token in sentence_tokenizer_output:
15         #print('\t{}'.format(token))
16         print(token)
17         f.write(token)
18         f.write("\n")
19     if not data:
20         print("end of file")
21         break
```



Florence, 29 January 2016 Press Release Fine dust, nitrogen dioxide and ozone improve overall Tuscan cities but there are still many critical issues Mal'Aria di città 2016 Regional data on air pollution 2015 ended under the banner of the smog emergency.

Most of the Tuscan cities "woke up" once again, around mid-December, with the urban background and traffic control units which recorded, almost continuously, for two weeks, exceedances of the PM10 above the limit value of 50 micrograms per cubic meter.

A limit that by law cannot be exceeded more than 35 times in a year.

These data emerged from the press conference for the presentation of the Legambiente Mal'aria 2016 dossier which took place this morning at the Giubbe Rosse literary café, in Piazza della Repubblica, in the presence of the regional president of Legambiente Fausto Ferruzza and the general director of ARPAT Maria Sargentini.

"The average situation has definitely improved and we can only be happy about it - declares Fausto Ferruzza, President of Legambiente Toscana - however, today that we are commenting on all in all flattering data, we cannot forget the smog emergency in December.

Many consecutive days of high pressure, fog and total absence of wind created a deadly cocktail of health alarms just a few weeks ago. The plan to prevent malaria in the city is known: care of the iron in local public transport, more cycle networks, more pedestrianizations in our ancient centres; domestic heating with the highest rate of innovation in industrial activities rigorous application of the European principle of the polluter pays, in view of a progressive reduction in the emission intensity of our production activities".

**Figura 3.4:** Dataset dopo aver effettuato la sentence tokenization.

### 3.3.3 Lowercase

Per migliorare il processo di Data Cleaning si è provveduto a trasformare tutti i dati testuali in minuscolo. Questa operazione risulta essere molto utile per l'eliminazione di distinzioni tra parole identiche nel loro significato, ma che risultano distinte graficamente per la presenza di lettere maiuscole. Ad esempio "Casa" e "casa", anche se sono la stessa parola, il nostro modello potrebbe trattarle come parole distinte.

```

1 from asyncio import write
2 from cgitb import text
3 import linecache
4 c=1
5 f = open("Prova.txt", mode='a', encoding='UTF-8')
6 while(True):
7     data = linecache.getline('datasetIngTokNoiseRimNoise.txt', c)
8     lowercased_text = data.lower()
9     print(lowercased_text)

```

```
10     f.write(lowercased_text)
11     c+=1
12     if not data:
13         print("end of file")
14         break
```

florence, 29 january 2016 press release fine dust, nitrogen dioxide and ozone improve overall tuscan cities but there are still many critical issues mal'aria di città 2016 regional data on air pollution 2015 ended under the banner of the smog emergency.

most of the tuscan cities "woke up" once again, around mid-december, with the urban background and traffic control units which recorded, almost continuously, for two weeks, exceedances of the pm10 above the limit value of 50 micrograms per cubic meter.

a limit that by law cannot be exceeded more than 35 times in a year.

these data emerged from the press conference for the presentation of the legambiente mal'aria 2016 dossier which took place this morning at the giubbe rosse literary café, in piazza della repubblica, in the presence of the regional president of legambiente fausto ferruzza and the general director of arpat maria sargentini.

"the average situation has definitely improved and we can only be happy about it - declares fausto ferruzza, president of legambiente toscana - however, today that we are commenting on all in all flattering data, we cannot forget the smog emergency in december.

many consecutive days of high pressure, fog and total absence of wind created a deadly cocktail of health alarms just a few weeks ago. the plan to prevent malaria in the city is known: care of the iron in local public transport, more cycle networks, more pedestrianizations in our ancient centres; domestic heating with the highest rate of innovation in industrial activities rigorous application of the european principle of the polluter pays, in view of a progressive reduction in the emission intensity of our production activities".

the smog emergency during the winter months is always triggered by fine dust, i.e.

pm10 and pm2.5. atmospheric particulate matter has for many years now been considered among the pollutants with the greatest impact on people's health, due to its "ability" to be easily inhaled by the respiratory system and due to the high concentrations that are recorded especially in urban environments.

**Figura 3.5:** Dataset dopo la trasformazione dei dati testuali in minuscolo.

### 3.3.4 Rimozione punteggiatura

Dopo aver trasformato tutto il testo in minuscolo si è provveduto a rimuovere la punteggiatura. I motivi per la quale la punteggiatura è stata rimossa sono molteplici:

- Riduzione del rumore: La presenza della punteggiatura può rappresentare un rumore nel dataset, per cui la sua rimozione può migliorare la precisione dell'analisi.
- Uniformità: Rimuovendo la punteggiatura, si ottiene un testo uniforme, in cui ogni parola è separata da uno spazio vuoto. Ciò semplifica l'elaborazione del testo.
- Riduzione della complessità: La presenza della punteggiatura può aumentare la complessità del dataset. Rimuovendo la punteggiatura, si ottiene un testo più semplice da elaborare.

Pertanto, la rimozione della punteggiatura può essere utile per migliorare la qualità e la precisione dell'analisi del testo.

```

1 from asyncore import write
2 from cgitb import text
3 import linecache
4 import re
5 c=1
6 f = open("Prova.txt", mode='a', encoding='UTF-8')
7 while(True):
8     data = linecache.getline('dataset.txt', c).strip()
9     punctuation = ""'!"#$%&'()*+,-./:;<=>?@[]^_`{|}~""
10    text_without_punctuation = "".join(i for i in data
11                                       if not i in punctuation)
12    print(text_without_punctuation)
13    f.write(text_without_punctuation)
14    f.write("\n")
15    c+=1
16    if not data:
17        print("end of file")
18        break

```



florence 29 january 2016 press release fine dust nitrogen dioxide and ozone improve overall tuscan cities but there are still many critical issues malaria di città 2016 regional data on air pollution 2015 ended under the banner of the smog emergency most of the tuscan cities "woke up" once again around middecember with the urban background and traffic control units which recorded almost continuously for two weeks exceedances of the pm10 above the limit value of 50 micrograms per cubic meter a limit that by law cannot be exceeded more than 35 times in a year these data emerged from the press conference for the presentation of the legambiente malaria 2016 dossier which took place this morning at the giubbe rosse literary café in piazza della repubblica in the presence of the regional president of legambiente fausto ferruzza and the general director of arpat maria sargentini the average situation has definitely improved and we can only be happy about it declares fausto ferruzza president of legambiente toscana however today that we are commenting on all in all flattering data we cannot forget the smog emergency in december many consecutive days of high pressure fog and total absence of wind created a deadly cocktail of health alarms just a few weeks ago the plan to prevent malaria in the city is known care of the iron in local public transport more cycle network more pedestrianizations in our ancient centres domestic heating with the highest rate of innovation in industrial activities rigorous application of the european principle of the polluter pays in view of a progressive reduction in the emission intensity of our production activities

**Figura 3.6:** Dataset dopo aver rimosso la punteggiatura.

### 3.3.5 Word tokenization

La Word tokenization è una tecnica di Data Cleaning che permette di suddividere il testo all'interno di un dataset in singole parole. Questa tecnica ci è stata molto utile per la rimozione dei noisy data.

```

1 from asyncore import write
2 from cgitb import text
3 import nltk
4 from nltk.tokenize import sent_tokenize
5 from nltk.tokenize import word_tokenize
6 import linecache
7
8 nltk.download('punkt')
9 c=1
10 f = open("Prova.txt", mode='a', encoding='UTF-8')
11 while(True):
12     data = linecache.getline('datasetIng.txt', c)

```

```
13     word_tokenizer_output = word_tokenize(data)
14     c+=1
15     for token in word_tokenizer_output:
16         #print('\t{}'.format(token))
17         print(token)
18         f.write(token)
19         f.write("\n")
20     if not data:
21         print("end of file")
22         break
```

```
florence
29
january
2016
press
release
fine
dust
nitrogen
dioxide
and
ozone
improve
overall
tuscan
cities
```

**Figura 3.7:** Dataset dopo aver effettuato la Word tokenization.

### 3.3.6 Rimozione dati rumorosi

I Noisy data (dati rumorosi) sono dati che contengono errori, imprecisioni o non sono rilevanti al problema in esame, possono essere causati da problemi nel processo di raccolta o di elaborazione. Pertanto potrebbero influenzare negativamente l'analisi e le previsioni, portando a risultati poco affidabili e poco rappresentativi. L'eliminazione dei dati rumorosi può quindi aiutare a:

- Migliorare la precisione delle analisi: i dati rumorosi possono distorcere i risultati delle analisi, dando luogo a conclusioni errate o poco affidabili.
- Migliorare la comprensione del dataset: l'eliminazione dei dati rumorosi può aiutare a evidenziare le relazioni e le tendenze presenti nei dati, rendendo più facile la loro interpretazione.
- Migliorare l'efficienza dell'analisi: l'eliminazione dei dati rumorosi può ridurre il tempo e le risorse necessarie per l'analisi, poiché si lavora con un dataset più pulito e di dimensioni più contenute.

```

1  from asyncio import write
2  from cgitb import text
3  import nltk
4  import re
5  from nltk.tokenize import sent_tokenize
6  from nltk.tokenize import word_tokenize
7  import linecache
8
9
10 c=1
11 f = open("Prova.txt", mode='a', encoding='UTF-8')
12 while(True):
13     data = linecache.getline('dataset.txt', c).rstrip("\n")
14
15     if(data.startswith("www")==True):
16         print(data)
17     elif(data.startswith("www")!=True):
18         f.write(data)
19         f.write("\n")
20     c+=1
21     if not data:
22         print("end of file")
23         break

```

## 3.4 Addestramento

Con l'avvenuta installazione del modello 774M e la creazione del dataset finale, si procede con il training del modello.

```
1 import gpt_2_simple as gpt2
2 import os
3 os.environ["CUDA_VISIBLE_DEVICES"] = "-1"
4 #nasconde i drive cuda a tensorflow per
5 #far si che venga utilizzata la cpu e non la gpu
6
7 sess = gpt2.start_tf_sess(threads=48)
8 #inizio sessione tensorflow creata gpt-2-simple
9
10
11
12 file_name = "Dataset\\datasetPdf.txt" #assegnazione file
13
14 gpt2.finetune(sess,
15               dataset=file_name,
16               model_name='774M',
17               steps=4000,
18               run_name='run1',
19               save_every=1000,
20               sample_every=8000,
21               restore_from='fresh'
22               )
```

Grazie alla libreria GPT-2-SIMPLE possiamo settare dei parametri per il finetuning. I parametri settati sono i seguenti:

- `model_name`: indica il modello che vogliamo utilizzare, nel nostro caso abbiamo utilizzato il modello 774M.
- `steps`: indica il numero di iterate di addestramento, nel nostro caso sono 4000.
- `save_every`: indica il numero di steps dopo il quale il modello memorizza un checkpoint.

- `sample_every`: indica ogni quanto il modello genererà degli esempi. Poiché ci siamo resi conto che nella versione 774M questo parametro provoca delle eccezioni, per cui è stato impostato ad 8000, per forzare la non generazione di esempi.
- `restore_from`: indica il checkpoint da cui si vuol partire per il fine-tuning. Questo parametro può avere due valori distinti: “fresh” e “latest”. Il parametro “fresh” ci indica che il modello partirà ad effettuare il training dall’inizio, mentre il parametro “latest” ci indica che il modello ripartirà dall’ultimo checkpoint memorizzato. Nel nostro caso, essendo il primo addestramento, abbiamo settato il valore “fresh”.

La sessione di addestramento del modello è durata circa 15 giorni, ottenendo una loss media di circa 0.03.

## 3.5 Generazione report

Una volta concluso l’addestramento del nostro dataset, si è passato alla generazione del report. Per generare il report è stato usato il seguente codice:

```

1 import gpt_2_simple as gpt2
2 import os
3 os.environ["CUDA_VISIBLE_DEVICES"] = "-1"
4
5 sess = gpt2.start_tf_sess()
6 gpt2.load_gpt2(sess, run_name='run1')
7
8 str = gpt2.generate(sess, run_name='run1', top_p=0.9,
9 top_k=10, prefix="the aqi", temperature=0.4,
10 length=200, return_as_list=True)[0]
11
12 print(str.replace("\n", " "))

```

Come si può notare dal codice sopra citato è stata utilizzata la funzione `generate` che può prendere in input diversi parametri.

Tra i parametri utilizzati abbiamo la `temperature` che, nel nostro caso, è stata im-

postata al valore 0.4. Pertanto, il grado di “fantasia” con cui il modello esegue gli accoppiamenti di parole risulta essere basso.

Mentre per i parametri `prefix` (che indica il prefisso con cui dovrebbe iniziare la parola) e `length`, i valori a loro assegnati, dipendono dal tipo di report che si vuole generare.

---

### Valutazione del modello

---

Un'operazione fondamentale durante la progettazione di un modello di machine learning è la sua valutazione. La valutazione di un modello di generazione di testo è importante perché ci permette di capire quanto bene il modello è in grado di creare dati simili a quelli di cui disponeva durante l'addestramento. Per questo motivo sono state poste le seguenti domande di ricerca:

**Q RQ<sub>1</sub>.** *Quali sono le performance del modello proposto in termini di correttezza sintattica e coerenza semantica dei report generati?*

**Q RQ<sub>2</sub>.** *Qual è il grado di comprensione dei report generati?*

In particolare, per la valutazione del nostro modello (descritto nel capitolo 3), sono state svolte due valutazioni distinte:

- Una prima valutazione riguardante la qualità del dataset ottenuto, grazie al processo di data cleaning. Quindi si valuterà quanto il dataset sia coerente con il contesto, andando a visualizzare attraverso un Word Cloud la differenza di parole-chiave tra il dataset prima e dopo il processo di data cleaning.







**Figura 4.2:** Word Cloud del dataset finale.

Come si evince dalla figura 4.1 e dalla figura 4.2, il nostro dataset, dopo aver effettuato il processo di data cleaning, è molto più conforme con il contesto ambientale. Infatti, se confrontiamo la figura 4.1 e la figura 4.2 possiamo notare come nella figura 4.1 le parole più importanti del dataset siano parole poco caratterizzanti per il nostro contesto, ad esempio: di, delle, per, della, mentre nella figura 4.2, ovvero la figura del dataset dopo il processo di data cleaning, le parole più caratterizzanti sono: data, air, pm, emissions, pollution. Questo ci fa comprendere come il processo di data cleaning svolto abbia influito sulla qualità generale del dataset, in termini di coerenza.

## 4.2 Valutazione delle frasi generate

Come già detto precedentemente, la valutazione delle frasi si è suddivisa in due parti. Una valutazione effettuata in maniera empirica, da esperti del dominio, che hanno analizzato le frasi “a mano” e una valutazione utilizzando un indice di leggibilità chiamato Gulpease. Il primo passo svolto è stato la generazione di 5 frasi con prefisso e lunghezza differenti. I parametri utilizzati sono i seguenti:

**Tabella 4.1:** Parametri per la generazione di frasi

Prefisso	Lunghezza
Today in Milan	100
Naples	100
The aqi	150
The aqi	200

La motivazione per la quale sono state scelte tale numero di frasi è perché la loro generazione è lunga per cui, per diminuire la messa in produzione, si è pensato di generarne solo 5. Di seguito vengono mostrate le frasi generate. (Per semplicità i report sono stati tradotti in italiano).

Frase 1: prefisso=Today in Milan, Lunghezza=100.

Oggi a Milano il pm è diminuito del 45% rispetto al triennio precedente, il trend del cambiamento climatico è notoriamente evidente anche nell'area industriale dove la produzione di materiale lapideo utilizzato per il cemento è la principale fonte di emissione rispetto al totale nazionale

**Figura 4.3:** prefisso=Today in Milan, Lunghezza=100.

Frase 2: prefisso= Naples, Lunghezza=100.

Napoli i valori medi annui a livello comunale o di altre aggregazioni sono ritenuti approssimativamente stabili nel tempo rendendo generalmente attendibili le stime ottenute anche in tempi recenti la metodologia di elaborazione dei dati è la stessa utilizzata nella qualità dell'ambiente urbano 2015

**Figura 4.4:** prefisso= Naples, Lunghezza=100.

Frase 3: prefisso= The aqi, Lunghezza=150.

L'aqi è stato oggetto di recenti studi volti a stimare il suo impatto sulla salute derivante dall'esposizione ad inquinanti atmosferici, ad esempio, wolverton et al 2013 la concentrazione di pm2.5 è fortemente legata alle condizioni meteorologiche della stagione invernale che possono essere più o meno favorevoli alla formazione di inquinanti o influenzarne la capacità di reazione formando altri composti

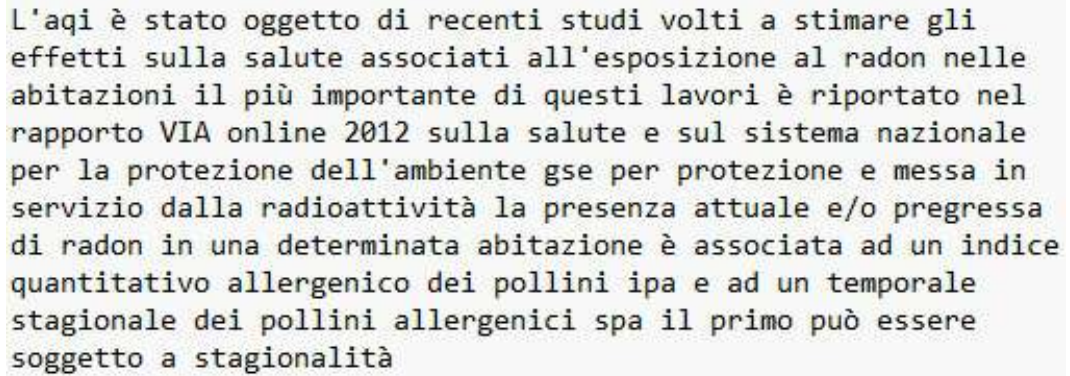
**Figura 4.5:** prefisso= The aqi, Lunghezza=150.

Frase 4: prefisso= The aqi, Lunghezza=200.

L'aqi è stato oggetto di recenti studi volti a stimare gli effetti sulla salute legati all'esposizione al radon nelle abitazioni il più importante di questi lavori è svolto a livello comunale nel caso della questione relativa all'esposizione al radon nelle abitazioni che riporta la concentrazione media di radon presente in ogni stato membro dell'unione europea per l'anno 2013 i risultati dell'indagine sono coerenti con quanto rilevato da un successivo studio di tominz et al 2003 che est

**Figura 4.6:** prefisso= The aqi, Lunghezza=200.

Frase 5: prefisso= The aqi, Lunghezza=200.



L'aqi è stato oggetto di recenti studi volti a stimare gli effetti sulla salute associati all'esposizione al radon nelle abitazioni il più importante di questi lavori è riportato nel rapporto VIA online 2012 sulla salute e sul sistema nazionale per la protezione dell'ambiente gse per protezione e messa in servizio dalla radioattività la presenza attuale e/o pregressa di radon in una determinata abitazione è associata ad un indice quantitativo allergenico dei pollini ipa e ad un temporale stagionale dei pollini allergenici spa il primo può essere soggetto a stagionalità

**Figura 4.7:** prefisso= The aqi, Lunghezza=200.

Una volta conclusa la generazione delle frasi si è passato alla valutazione di esse. Grazie alla valutazione effettuata da esperti del dominio, i quali hanno validato il senso della frase, ci siamo resi conto che il risultato delle frasi generate sono abbastanza soddisfacenti, sia in termini di correttezza sintattica che di coerenza semantica, ma non prive di difetti. Le parole utilizzate fanno intendere che il modello ha ben imparato il vocabolario specifico di parole che vengono utilizzate prettamente nell'ambito dell'ambiente e dell'inquinamento. Le frasi hanno abbastanza senso logico e si legano bene tra di loro, ma si può notare, dall'analisi effettuata, come il parametro length possa avere un impatto sulle frasi generate, infatti, più sarà grande il valore di questo parametro e più il modello avrà difficoltà a generare una frase che sia coerente e corretta. Dopo la valutazione effettuata dagli esperti del dominio si è passato all'utilizzo dell'Indice di Gulpease. L'indice di Gulpease è un indice di leggibilità del testo, ovvero si occupa di valutare il grado di comprensione di un testo. La sua peculiarità è quella di essere in grado non solo di misurare la leggibilità di un testo in valore assoluto, ma anche rispetto ai vari livelli di scolarizzazione. Per fare ciò l'indice di Gulpease considera due variabili linguistiche: la lunghezza della parola e la lunghezza della frase rispetto al numero delle lettere.

Di seguito è riportata la formula utilizzata:

$$89 + \frac{300 * (\text{numero delle frasi}) - 10 * (\text{numero delle lettere})}{\text{numero delle parole}}$$

La scala di leggibilità va da 100 a 0, dove 100 indica la leggibilità più alta e 0 quella più bassa.

**Tabella 4.2:** Scala di leggibilità

Valore	Significato
< 80	testo difficile da leggere per chi ha la licenza elementare
< 60	testo difficile da leggere per chi ha la licenza media
< 40	testo difficile da leggere per chi ha un diploma superiore

I risultati ottenuti sono i seguenti:

**Tabella 4.3:** Risultati delle frasi ottenute tramite l'indice di Gulpease

Frase n.	Punteggio ottenuto	Riferimento
1	40	Figura4.3
2	36	Figura4.4
3	35	Figura4.5
4	39	Figura4.6
5	39	Figura4.7

Analizzando la tabella soprastante, ci accorgiamo che i risultati ottenuti tramite l'indice di Gulpease sono abbastanza bassi, infatti, ad eccetto della frase numero 1

(Figura4.3, prefisso=Today in Milan, Lunghezza=100.) tutte le altre frasi hanno un indice di leggibilità minore di 40, pertanto risultano difficili da leggere per chi ha un diploma superiore. In conclusione, alla luce della valutazione svolta, possiamo rispondere alle due domande di ricerca preposte precedentemente (ad inizio capitolo 4).

🔗 **Answer to RQ<sub>1</sub>.** Dai risultati ottenuti grazie alla valutazione degli esperti del dominio, possiamo affermare che le frasi generate sono abbastanza soddisfacenti, sia in termini di correttezza sintattica che di coerenza semantica. Infatti, le frasi generate hanno abbastanza senso logico e si legano bene tra di loro.

🔗 **Answer to RQ<sub>2</sub>.** Il grado di comprensione delle frasi generate risulta basso, infatti, la maggior parte delle frasi risulta difficile da leggere per chi ha un diploma superiore, poiché la frase generata ha un punteggio inferiore a 40.

Pertanto, possiamo affermare che i risultati ottenuti non rispettano completamente gli obiettivi preposti dall'azienda. Questo è dovuto principalmente alla qualità dei dati di input, di fatti i dati erano eterogeni tra loro e in aggiunta, c'era una quantità elevata di dati "rumorosi". Alla luce di ciò le frasi generate non sono state ritenute sufficienti per il loro obiettivo principale, ma comunque possono fornire una base per i possibili sviluppi futuri.

---

### Conclusioni e sviluppi futuri

---

L'obiettivo del lavoro svolto si è incentrato sullo sviluppo di un sistema automatico ed intelligente in grado di generare dei piccoli report ambientali, che descrivono in maniera accurata lo stato della qualità dell'aria ed eventi storici di una determinata provincia o regione, partendo da dati reali o proprietari. Con l'utilizzo del sistema implementato si consente all'utenza di Sense Square di recuperare informazioni inerenti allo stato ambientale attraverso le frasi generate dal modello. Siamo partiti dall'analisi di varie documentazioni per una comprensione del dominio applicativo e del dominio delle soluzioni, che ci ha permesso di acquisire diverse conoscenze relative al concetto di NLG e delle diverse architetture di rete utilizzate in quest'ambito, inoltre, questa analisi è servita anche per la costruzione del secondo capitolo di background. Grazie allo studio iniziale ed alla comprensione degli obiettivi dall'azienda, sono state fatte delle scelte sulle tecnologie da utilizzare per l'obiettivo principale, come ad esempio l'uso di GPT-2 e non di GPT-3, a causa del fatto che l'azienda aveva bisogno di un sistema proprietario e compatibile con l'hardware a disposizione. Una volta conclusa la fase di studio iniziale e della scelta delle tecnologie si è passato allo sviluppo del sistema, dove se ne parla nel terzo capitolo. La maggiore delle difficoltà presentatesi è stata la costruzione del dataset, poiché i dati a disposizione erano pochi e molto eterogenei tra loro. Di fatti il dataset ri-

sultante è di piccole dimensioni. Con la valutazione del modello, dove se ne parla nel quarto capitolo, ci si è resi conto che le frasi generate sono abbastanza coerenti semanticamente, di fatti si legano bene tra loro ed hanno senso logico ma sono poco leggibili per chi non ha un livello di scolarizzazione elevato, in più peccano di diversificazione tra parole quando esse iniziano ad assumere una dimensione abbastanza grande. Apprendendo le principali problematiche del modello si è analizzati quali fossero i possibili sviluppi futuri. Come possibili sviluppi futuri, si potrebbe ambire a:

- Un ampliamento del dataset cercando di ottenere dati che non siano tra loro troppo eterogeni.
- Ulteriori sessioni di tuning dei parametri e degli iperparametri.
- L'utilizzo di altre tecniche di data cleaning in modo da migliorare la qualità dei dati all'interno del dataset.



---

## Bibliografia

---

- [1] "Immagini della piattaforma sensesquare," 2023, controllata il 28-02-2023. [Online]. Available: <https://square.sensesquare.eu/> (Citato alle pagine iii, 2 e 3)
- [2] Datecon, "Digital voice assistant for data retrieval in reporting," 2022, controllata il 28-02-2023. [Online]. Available: <https://www.datecon.com/en/journal/digital-voice-assistant-data-retrieval-reporting> (Citato alle pagine iii e 7)
- [3] SAP, "Che cos'è il machine learning?" controllata il 28-02-2023. [Online]. Available: <https://www.sap.com/italy/insights/what-is-machine-learning.html> (Citato alle pagine iii e 11)
- [4] Appuntioss, "Sistema nervoso - i neuroni," controllata il 28-02-2023. [Online]. Available: <https://www.appuntioss.it/anatomia-corpo-sistema-nervoso-i-neuroni/> (Citato alle pagine iii e 12)
- [5] C. D. Nardo, "Cos'è l'intelligenza artificiale?" 2021, controllata il 28-02-2023. [Online]. Available: <https://deltalogix.blog/2021/03/10/intelligenza-artificiale/> (Citato alle pagine iii e 13)
- [6] L. Zanotti, "Reti neurali e deep learning: applicazioni commerciali in continua crescita," 2020, controllata il 28-02-2023. [Online]. Available: <https://www.zerounoweb.it/cio-innovation/>

- reti-neurali-e-deep-learning-applicazioni-commerciali-in-continua-crescita/  
(Citato alle pagine iii e 13)
- [7] Wikipedia, "File:artificialneuronmodel english.png — wikipedia, l'enciclopedia libera," 2021, controllata il 28-02-2023. [Online]. Available: [https://it.wikipedia.org/wiki/File:ArtificialNeuronModel\\_english.png](https://it.wikipedia.org/wiki/File:ArtificialNeuronModel_english.png) (Citato alle pagine iii e 14)
- [8] A. Mittal, "Understanding rnn and lstm," 2019, controllata il 28-02-2023. [Online]. Available: <https://aditi-mittal.medium.com/understanding-rnn-and-lstm-f7cdf6dfc14e> (Citato alle pagine iii e 22)
- [9] S. Dobilas, "Lstm recurrent neural networks — how to teach a network to remember the past," Available: <https://towardsdatascience.com/lstm-recurrent-neural-networks-how-to-teach-a-network-to-remember-the-past-55e54c2ff22e>, 2022, [Online], Controllata il 28-02-2023. (Citato alle pagine iii e 23)
- [10] A. Ranjan, "Image captioning with an end-to-end transformer network," 2022, controllata il 28-02-2023. [Online]. Available: <https://python.plainenglish.io/image-captioning-with-an-end-to-end-transformer-network-8f39e1438cd4> (Citato alle pagine iii e 24)
- [11] B. K, "Bert transformers for natural language processing," 2022, controllata il 28-02-2023. [Online]. Available: <https://blog.paperspace.com/bert-natural-language-processing/> (Citato alle pagine iii e 25)
- [12] J. Alammar, "The illustrated gpt-2 (visualizing transformer language models)," 2019, controllata il 28-02-2023. [Online]. Available: <https://jalammar.github.io/illustrated-gpt2/> (Citato alle pagine iii, iv, 26, 27, 28, 29, 30 e 31)
- [13] M. Woolf, "gpt-2-simple," 2021, controllata il 28-02-2023. [Online]. Available: <https://github.com/minimaxir/gpt-2-simple> (Citato alle pagine v, 32, 33 e 36)
- [14] Wikipedia, "Elaborazione del linguaggio naturale — wikipedia, l'enciclopedia libera," 2022, controllata il 28-02-2023. [Online]. Available: [https://it.wikipedia.org/wiki/Elaborazione\\_del\\_linguaggio\\_naturale](https://it.wikipedia.org/wiki/Elaborazione_del_linguaggio_naturale) (Citato alle pagine 6 e 7)

- [15] T. F. W. Ben Lutkevich, "Definition natural language processing (nlp)," 2023, controllata il 28-02-2023. [Online]. Available: <https://www.techtarget.com/searchenterpriseai/definition/natural-language-processing-NLP> (Citato a pagina 6)
- [16] A. V. D. RES, "La costruzione di un modello di natural language processing: Dalla raccolta alla pulizia dei dati," Available: <https://res-group.eu/articoli/la-costruzione-di-un-modello-di-natural-language-processing-dalla-raccolta-alla-pulizia-dei-dati>, 2021, [Online], Controllata il 28-02-2023. (Citato a pagina 6)
- [17] ihal, "Che cos'è la comprensione del linguaggio naturale nlu?" 2020, controllata il 28-02-2023. [Online]. Available: <https://ihal.it/che-cose-la-comprensione-del-linguaggio-naturale-nlu/> (Citato alle pagine 7 e 8)
- [18] P. Dotti, "Natural language generation, come andare oltre i chatbot e gli assistenti vocali," 2021, controllata il 28-02-2023. [Online]. Available: <https://www.ai4business.it/intelligenza-artificiale/natural-language-generation-oltre-i-chatbot-e-gli-assistenti-vocali/> (Citato alle pagine 8 e 17)
- [19] A. AWS, "Cos'è il deep learning?" 2023, controllata il 28-02-2023. [Online]. Available: <https://aws.amazon.com/it/what-is/deep-learning/> (Citato alle pagine 9, 10, 11, 13, 16 e 17)
- [20] M. R. Carbone, "Transfer learning, cos'è, come funziona e applicazioni," 2022, controllata il 28-02-2023. [Online]. Available: <https://www.ai4business.it/intelligenza-artificiale/transfer-learning-cose-come-funziona-e-applicazioni/> (Citato a pagina 10)
- [21] F. R. Mashrur, "What is the difference between transfer learning vs fine tuning vs. learning from scratch?" Available: <https://www.researchgate.net/post/What-is-the-difference-between-Transfer-Learning-vs-Fine-Tuning-vs-Learning-from->

- scratch#:~:text=Training%20from%20scratch%20means%20that,a%20lot%20of%20computational%20power., 2020, [Online], Controllata il 28-02-2023. (Citato a pagina 10)
- [22] H. Ampadu, "Dropout in deep learning," 2021, controllata il 28-02-2023. [Online]. Available: <https://ai-pool.com/a/s/dropout-in-deep-learning> (Citato a pagina 10)
- [23] R. Kundu, "Everything you need to know about few-shot learning," 2022, controllata il 28-02-2023. [Online]. Available: <https://blog.paperspace.com/few-shot-learning/> (Citato a pagina 10)
- [24] Wikipedia, "Rete neurale — wikipedia, l'enciclopedia libera," 2021, controllata il 28-02-2023. [Online]. Available: [https://it.wikipedia.org/wiki/Rete\\_neurale](https://it.wikipedia.org/wiki/Rete_neurale) (Citato alle pagine 12 e 13)
- [25] N. Canu, "Neurone," 2010, controllata il 28-02-2023. [Online]. Available: [https://www.treccani.it/enciclopedia/neurone\\_res-f6daa336-9b53-11e1-9b2f-d5ce3506d72e\\_%28Dizionario-di-Medicina%29/](https://www.treccani.it/enciclopedia/neurone_res-f6daa336-9b53-11e1-9b2f-d5ce3506d72e_%28Dizionario-di-Medicina%29/) (Citato a pagina 12)
- [26] Wikipedia, "Rete neurale artificiale — wikipedia, l'enciclopedia libera," 2023, controllata il 28-02-2023. [Online]. Available: [https://it.wikipedia.org/wiki/Rete\\_neurale\\_artificiale](https://it.wikipedia.org/wiki/Rete_neurale_artificiale) (Citato alle pagine 13, 14 e 15)
- [27] C. Casellato, "Qual è la differenza tra funzione di attivazione e funzione di trasferimento in una rete neurale?" Available: <https://it.quora.com/Qual-%C3%A8-la-differenza-tra-funzione-di-attivazione-e-funzione-di-trasferimento-in-una-rete-neurale>, 2019, [Online], Controllata il 28-02-2023. (Citato a pagina 14)
- [28] NetAi, "Rguida rapida alle funzioni di attivazione nel deep learning," 2021, controllata il 28-02-2023. [Online]. Available: <https://netai.it/guida-rapida-alle-funzioni-di-attivazione-nel-deep-learning/#page-content> (Citato a pagina 15)

- [29] Wikipedia, “Modèle de langage — wikipedia, l’enciclopedia libera,” 2023, controllata il 28-02-2023. [Online]. Available: [https://fr.wikipedia.org/wiki/Mod%C3%A8le\\_de\\_langage](https://fr.wikipedia.org/wiki/Mod%C3%A8le_de_langage) (Citato alle pagine 17 e 20)
- [30] —, “Traduzione automatica — wikipedia, l’enciclopedia libera,” 2022, controllata il 28-02-2023. [Online]. Available: [https://it.wikipedia.org/wiki/Traduzione\\_automatica](https://it.wikipedia.org/wiki/Traduzione_automatica) (Citato a pagina 17)
- [31] —, “Spell checker — wikipedia, l’enciclopedia libera,” 2023, controllata il 28-02-2023. [Online]. Available: [https://en.wikipedia.org/wiki/Spell\\_checker](https://en.wikipedia.org/wiki/Spell_checker) (Citato a pagina 17)
- [32] —, “N-gramma — wikipedia, l’enciclopedia libera,” 2020, controllata il 28-02-2023. [Online]. Available: <https://it.wikipedia.org/wiki/N-gramma> (Citato alle pagine 18 e 19)
- [33] E. Santus, “I limiti dei modelli linguistici e la sfida di ai21 labs,” 2022, controllata il 28-02-2023. [Online]. Available: <https://www.notizie.ai/i-limiti-dei-modelli-linguistici-e-la-sfida-di-ai21-labs/> (Citato alle pagine 18 e 19)
- [34] E. Giannini, “Modello linguistico con n-grammi,” 2020, controllata il 28-02-2023. [Online]. Available: <https://enricogiannini.com/15/modello-linguistico-con-n-grammi/> (Citato alle pagine 18 e 19)
- [35] Wikipedia, “Rete neurale feed-forward — wikipedia, l’enciclopedia libera,” 2022, controllata il 28-02-2023. [Online]. Available: [https://it.wikipedia.org/wiki/Rete\\_neurale\\_feed-forward](https://it.wikipedia.org/wiki/Rete_neurale_feed-forward) (Citato alle pagine 19 e 20)
- [36] —, “Rete neurale ricorrente — wikipedia, l’enciclopedia libera,” 2022, controllata il 28-02-2023. [Online]. Available: [https://it.wikipedia.org/wiki/Rete\\_neurale\\_ricorrente](https://it.wikipedia.org/wiki/Rete_neurale_ricorrente) (Citato alle pagine 19 e 22)
- [37] —, “Long short-term memory — wikipedia, l’enciclopedia libera,” 2023, controllata il 28-02-2023. [Online]. Available: [https://en.wikipedia.org/wiki/Long\\_short-term\\_memory](https://en.wikipedia.org/wiki/Long_short-term_memory) (Citato alle pagine 20 e 23)

- [38] —, “Transformer (machine learning model) — wikipedia, l’enciclopedia libera,” 2023, controllata il 28-02-2023. [Online]. Available: [https://en.wikipedia.org/wiki/Transformer\\_\(machine\\_learning\\_model\)](https://en.wikipedia.org/wiki/Transformer_(machine_learning_model)) (Citato alle pagine 20 e 25)
- [39] P. Sharma, “Feedforward neural network: Its layers, functions, and importance,” 2022, controllata il 28-02-2023. [Online]. Available: <https://www.analyticsvidhya.com/blog/2022/01/feedforward-neural-network-its-layers-functions-and-importance/> (Citato a pagina 20)
- [40] geeksforgeeks, “Introduction to recurrent neural network,” 2022, controllata il 28-02-2023. [Online]. Available: <https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/> (Citato a pagina 22)
- [41] S. K. T, “Natural language processing – sentiment analysis using lstm,” 2022, controllata il 28-02-2023. [Online]. Available: <https://www.analyticsvidhya.com/blog/2021/06/natural-language-processing-sentiment-analysis-using-lstm/> (Citato a pagina 23)
- [42] Nvidia, “What is a transformer model?” 2022, controllata il 28-02-2023. [Online]. Available: <https://blogs.nvidia.com/blog/2022/03/25/what-is-a-transformer-model/> (Citato a pagina 25)
- [43] R. Jagtap, “Openai gpt: Generative pre-training for language understanding,” 2020, controllata il 28-02-2023. [Online]. Available: <https://medium.com/dataseries/openai-gpt-generative-pre-training-for-language-understanding-bbbdb42b7ff4> (Citato a pagina 26)
- [44] Wikipedia, “Gpt-3 — wikipedia, l’enciclopedia libera,” 2023, controllata il 28-02-2023. [Online]. Available: <https://it.wikipedia.org/wiki/GPT-3> (Citato a pagina 26)

---

## Ringraziamenti

---

Quando iniziai il mio percorso universitario, non sapevo cosa aspettarmi, mi ripetevo molto spesso: “E’ la decisione giusta? Riuscirò a portare al termine questo obbiettivo?”. Erano trascorsi già due anni dal diploma e la paura di perdere altro tempo mi bloccava, ma decisi di affrontare le mie insicurezze e più andavo avanti più queste svanivano, così esame dopo esame, esperienza dopo esperienza, i tre anni sono trascorsi e finalmente il giorno della mia laurea è giunto. Il percorso da me svolto è stato molto arduo, ma per fortuna accanto a me ho avuto persone che mi hanno dato forza di reagire nei momenti più difficili. Volevo innanzitutto ringraziare il professore Fabio Palomba che si è dimostrato subito entusiasta dell’argomento scelto per la mia tesi, fornendomi tutto l’aiuto necessario per superare questo ultimo passo del mio percorso. Ringrazio vivamente l’intero personale di Sense Square SRL, che mi ha accolto all’interno dell’azienda facendomi sentire subito a mio agio. Un ringraziamento alla mia famiglia, a mio padre e mia madre che mi hanno sempre sostenuto e spronato ad inseguire ogni mio sogno, a mia sorella Rita, della quale vado molto fiero, per il coraggio che ha sempre dimostrato nell’affrontare la vita, insegnandomi a non arrendermi alle prime avversità. Ringrazio i miei nonni e i miei zii per tutti i momenti trascorsi insieme e per l’affetto che mi hanno dato. Ora vorrei ringraziare i miei amici di sempre come Francesco, Paolino, Dario, Domenico, Mariachiara, Claudio ecc., con la quale ho condiviso molte delle mie esperienze e vissuto tanti bei momenti, dalle vacanze fatte insieme, le giornate a parlare del più

---

e del meno alle serate passate a giocare al pro-club. Quei bei momenti, come quelli che ci saranno in futuro, li porterò sempre con me. Un ringraziamento al mio amico Emmanuel, che mi è stato accanto nel momento più buio che vissuto, sostenendomi e incoraggiandomi a non mollare, devo molto a te Manu e sappi che per qualsiasi cosa io ci sarò sempre per te, ti voglio tanto bene amico mio. Ringrazio i miei compagni di università Simone, Rocco, Antonio e Francesco, componenti del team Saarf e Raaf\_gaming. In questi tre anni ne abbiamo vissute tante di esperienze, facendoci sempre forza l'un l'altro e creando tanti bei ricordi, come i progetti svolti insieme (dove potevi sentire Francesco e Antonio imprecare quando c'era qualcosa che non andava), beh amici miei se oggi sono qui lo devo anche a voi. In conclusione, volevo fare un ringraziamento speciale a mia nonna Carmela, a cui dedico la mia tesi. Nonna, ho trascorso molte notti insonni a pensare a cosa scriverti, ma trovare le parole giuste per ringraziarti di ciò che tu sei stata per me è quasi impossibile. Sei stata molte cose, un'amica, una confidente, la mia più grande ammiratrice. Ti ricordi quando venivo a casa tua e parlavamo dei miei esami svolti? Oppure quando mi dicevi di togliermi gli orecchini perché non ti piacevano. Ad oggi, non sai cosa darei per sentirti dire nuovamente: "Togliti sti cosi dalle orecchie che non mi piacciono", da quando non ci sei più tutto è cambiato, eri colei che dava forza a tutti, anche nell'ultimo periodo, quando a causa della tua malattia, non riuscivi più a stare in piedi. Mi ricordo le giornate passate insieme, quando ti venivo a trovare, aprivo la porta ed esordivo con la mia solita frase: "A nò è arrivat 'o sol tuojo" e tu mi guardavi e sorridevi. In quel momento io ero felicissimo perché, anche se per un'istante, riuscivo a farti distogliere l'attenzione dal dolore che provavi. Nonna, in questo giorno finalmente ho realizzato il tuo desiderio e anche se oggi non sei qui con noi, so che sei fiera di me.