ADVERSARIAL ATTACKS: ANALISI COMPARATIVA TRA MODELLI NEURALI E NON NEURALI IN NLP

Frizziero Alex
Dipartimento di Matematica e Informatica
Università degli Studi di Ferrara
Ferrara, IT
alex.frizziero@edu.unife.it

ABSTRACT

Gli adversarial attack rappresentano una sfida significativa nel campo del Natural Language Processing (NLP), evidenziando vulnerabilità critiche nei sistemi di elaborazione del linguaggio naturale. Questo studio presenta un'analisi comparativa dell'efficacia degli attacchi avversari su due diverse architetture: i classificatori tradizionali e i modelli neurali. La ricerca si concentra sulla valutazione delle principali metodologie di attacco, fornendo prima una panoramica generale del problema, ed esaminando le differenze nelle strategie di perturbazione e dei loro effetti su sistemi con architetture fondamentalmente diverse.

1 INTRODUZIONE

Negli ultimi anni, i sistemi di Intelligenza Artificiale (IA) e Machine Learning (ML) hanno registrato progressi e diffusione significativi, arrivando a trasformare e ridefinire completamente alcuni dei principali settori applicativi. In particolare, nel campo del Natural Language Processing (NLP), l'avvento dei modelli neurali profondi e, più recentemente, dei Large Language Models (LLMs), ha rivoluzionato il panorama delle applicazioni basate sul linguaggio naturale, come sistemi di traduzione automatica o Q&A. Tuttavia, parallelamente a questi avanzamenti, è emersa una criticità fondamentale: la vulnerabilità di questi sistemi agli adversarial attack. Gli adversarial attack rappresentano una classe di tecniche finalizzate a compromettere il funzionamento dei modelli di machine learning attraverso una manipolazione degli input, strategica ed impercettibile dal punto di vista umano. Nel contesto NLP, questi attacchi assumono caratteristiche peculiari dovute alla natura discreta e strutturata del linguaggio naturale. A differenza dei domini continui come la computer vision, dove le perturbazioni possono essere applicate modificando direttamente i valori dei pixel, nel NLP le modifiche devono preservare la sintassi e la semantica del testo, rendendo il processo di generazione degli esempi avversari particolarmente complesso.

In questo scenario, lo studio degli adversarial attack si rivela fondamentale per molteplici ragioni. Come accennato in precedenza, la crescente adozione di modelli in applicazioni sensibili rende imprescindibile garantire la loro robustezza e affidabilità. Idealmente, comprendere le vulnerabilità esistenti permette di sviluppare strategie di difesa efficaci e migliorare la resilienza di questi sistemi, riducendo il rischio di manipolazioni e comportamenti indesiderati. Oltre a questo, gli attacchi avversari sollevano anche questioni fondamentali riguardo all'interpretabilità e all'allineamento umano dei modelli. Dal punto di vista dell'explainability, la capacità di un modello di produrre output errati o ingannevoli a seguito di perturbazioni minime dimostra la fragilità delle sue rappresentazioni interne e la difficoltà nel fornire spiegazioni affidabili delle proprie decisioni, specie nelle Deep Neural Networks (DNNs). Come detto da (Goodfellow et al., 2015) in *Explaining and Harnessing Adversarial Examples*:

> *"The existence of adversarial examples suggests that being able to explain the training data or even being able to correctly label the test data does not imply that our models truly understand the tasks we have asked them to perform."*

Inoltre, lo human alignment, ovvero la coerenza tra il comportamento del modello e le aspettative umane, risulta compromesso in quanto le sue risposte possono essere facilmente manipolate in modi che confondono l'utente o che portano ad azioni indesiderate o dannose.

Il presente lavoro si propone di condurre un'analisi sistematica e comparativa degli adversarial attack, fornendo una panoramica del problema e delle definizioni rilevanti, prima in un contesto generale per poi focalizzarsi nel campo del NLP. Ci occuperemo di evidenziare le principali metodologie di attacco e difesa con le relative sfide e limitazioni. Nei capitoli successivi verrà proposto uno studio comparativo sull'efficacia degli attacchi in un modello non neurale di classificazione del testo e un modello basato su deep neural network, prestando particolare attenzione alle differenze e alle relative implicazioni sulla robustezza dei modelli. L'elaborato si conclude cercando di proporre riflessioni sul futuro degli adversarial attack nel contesto dell'evoluzione dei modelli linguistici e delle sfide emergenti nel campo dell'NLP. Attraverso questo studio, si mira a contribuire alla comprensione delle vulnerabilità nei modelli e a promuovere lo sviluppo di sistemi più robusti e allineati con le esigenze umane.

Nello specifico il documento è organizzato come segue: vengono forniti i fondamenti teorici necessari a comprendere il problema degli adversarial attack nel Capitolo 2. Nel Capitolo 3 il fenomeno viene analizzato nel contesto del Natural Language Processing. I casi applicativi con le rispettive caratteristiche vengono proposti nel Capitolo 4. Segue l'analisi comparativa e critica dei due modelli nel Capitolo 5. La ricerca si chiude con le considerazioni finali nel Capitolo 6.

2 Fondamenti Teorici degli Adversarial Attack

La comprensione degli adversarial attack richiede innanzitutto una chiara definizione della teoria soggiacente. Attraverso l'analisi degli attori, delle diverse tipologie di attacco, delle metriche di valutazione e delle principali sfide, emergerà un quadro completo delle attuali conoscenze in questo campo, ponendo le basi per i capitoli successivi dove verranno esplorate implementazioni specifiche e casi di studio.

2.1 Tassonomia Generale

Nel contesto del machine learning, un adversarial attack rappresenta un tentativo deliberato di manipolare il comportamento di un modello attraverso la creazione di input specificamente progettati per indurre errori nelle predizioni. Questo processo si basa sulla generazione di *adversarial example*, ovvero input modificati che, pur mantenendo caratteristiche semantiche simili all'input originale, causano classificazioni errate da parte del modello. Per formalizzare questo concetto, consideriamo un modello di machine learning come una funzione

$$f : X \to Y$$

che mappa uno spazio di input X in uno spazio di output Y. Un adversarial example può essere matematicamente definito come una perturbazione dell'input originale x tale che

$$x' = x + \delta$$

dove δ rappresenta la perturbazione. Questa perturbazione deve soddisfare vincoli specifici: deve essere sufficientemente piccola da preservare le caratteristiche essenziali dell'input originale (vincolo di perturbazione), deve causare una classificazione errata da parte del modello (condizione di successo), e deve mantenere la validità semantica dell'input nel dominio di applicazione. Se aiuta, si può pensare alla perturbazione come al rumore o alla modifica calcolata e applicata all'input, mentre l'adversarial example rappresenta il risultato finale, cioè il dato alterato che include la perturbazione.

Per comprendere meglio non solo il problema ma anche i reali rischi degli attacchi forniamo qualche semplice esempio. Un esempio emblematico di adversarial attack nel contesto della visione artificiale riguarda la manipolazione di un segnale di stop per ingannare un veicolo a guida autonoma. In questo scenario, un attaccante potrebbe applicare piccole modifiche al segnale, impercettibili per l'occhio umano ma sufficienti a confondere il sistema di riconoscimento del veicolo. In *Practical Black-Box Attacks against Machine Learning* (Papernot et al., 2017) i ricercatori hanno dimostrato che aggiungendo piccoli adesivi specificamente posizionati su un cartello stradale, un modello di computer vision potrebbe erroneamente classificarlo come un limite di velocità invece che come un

segnale di stop. Le implicazioni per la sicurezza in questo scenario risultano evidenti, pertanto vi è la necessità di sviluppare modelli più robusti, capaci di riconoscere e mitigare tali attacchi attraverso le opportune difese. Un ulteriore esempio è lo scenario descritto dai ricercatori in *Adversarial Interaction Attack: Fooling AI to Misinterpret Human Intentions* (Koren et al., 2021), nel quale si immagina un contesto di interazione tra uomo e agente robotico, introducendo una tipologia di attacchi chiamata *Adversarial Interaction Attack*. L'AIA mira a ingannare l'agente AI reattore facendogli credere che l'attore stia compiendo un'azione specifica diversa, apportando piccole modifiche alle posizioni delle articolazioni dell'attore o agli angoli tra le articolazioni. Di conseguenza, l'agente reattore risponderà eseguendo la reazione mirata dall'attacco. Le motivazioni dietro l'esecuzione di un attacco possono essere molteplici e dipendono dal contesto e dagli obiettivi dell'attaccante. Indurre il modello a fare previsioni errate o a comportarsi in modo inatteso, oppure rubare modelli o i dati utilizzati per l'addestramento sono solo alcuni dei possibili moventi. Mentre i cyberattacchi del mondo reale sono tipicamente dolosi, molti adversarial attack sono il risultato di sforzi di ricerca per evidenziare le vulnerabilità dei modelli di apprendimento automatico e la necessità di difese più forti per la sicurezza.

2.2 Tipologie di Minaccia

Gli adversarial attack possono essere categorizzati secondo diverse dimensioni, ognuna delle quali evidenzia aspetti specifici della metodologia di attacco e delle sue implicazioni pratiche. Una prima fondamentale distinzione riguarda il livello di accesso e conoscenza del modello target. Gli attacchi *white-box* rappresentano lo scenario in cui l'attaccante ha accesso completo all'architettura del modello, ai suoi parametri e ai gradienti. Questa categoria include tecniche come il *Fast Gradient Sign Method* (FGSM) e il *Projected Gradient Descent* (PGD), che sfruttano direttamente le informazioni sul gradiente del modello per ottimizzare le perturbazioni. FGSM, in particolare, opera calcolando il gradiente della *loss function* rispetto all'input e utilizzando il suo segno per generare perturbazioni nella direzione che massimizza l'errore del modello. Gli attacchi *black-box*, d'altra parte, operano in uno scenario dove l'attaccante ha accesso limitato al modello, tipicamente solo attraverso la sua interfaccia di input/output. Questi attacchi richiedono strategie più sofisticate, come l'addestramento di modelli surrogati o l'utilizzo di tecniche basate su *query* che esplorano iterativamente lo spazio delle perturbazioni attraverso tentativi successivi.

Un'altra distinzione fondamentale riguarda l'obiettivo dell'attacco. Gli attacchi *targeted* mirano a indurre il modello a produrre una specifica predizione errata; in questo caso, l'attaccante ha un obiettivo preciso in mente e vuole che il modello produca un output specifico e predeterminato. Nel caso complementare, gli attacchi *untargeted* si limitano a causare qualsiasi classificazione diversa da quella corretta; in altre parole, l'attaccante vuole solo che il modello produca un output errato, senza preoccuparsi di quale sia l'output specifico. Nel contesto della classificazione binaria, un attacco untargeted equivale a un attacco targeted, poiché l'unica alternativa alla classe corretta è l'altra classe disponibile. Questa distinzione ha importanti implicazioni sia per la complessità dell'attacco sia per le sue potenziali applicazioni malevole.

Un'altra categoria riguarda la granularità, ovvero il livello di dettaglio a cui vengono apportate le perturbazioni all'input. In altre parole, definisce su quale unità di base viene manipolato l'input per ingannare il modello. Nel contesto di origine di applicazione degli adversarial attack, gli elementi attaccati sono tipicamente i pixel di un'immagine; tuttavia, con l'espansione dello studio a campi come il Natural Language Processing, le manipolazioni coinvolgono caratteri, parole o addirittura frasi dell'input. Ulteriori distinzioni sono possibili in applicazioni di NLP, le cui implicazioni verranno trattate ed approfondite nel capitolo 3.

2.3 Misurazioni e Valutazioni

Gli attacchi avversari sono progettati per degradare le prestazioni dei modelli. Pertanto, la valutazione dell'efficacia dell'attacco si basa sulle metriche di prestazione dei diversi *task*. In altri termini, l'analisi degli adversarial attack si concentra principalmente sulla misurazione della performance dei modelli in presenza di tali attacchi, sia prima che dopo l'implementazione di meccanismi di difesa. La metrica più immediata è l'*Attack Success Rate* (ASR), che quantifica la percentuale di casi in cui l'attacco riesce a indurre una classificazione errata; nello specifico, indica quante volte gli esempi creati per attaccare il modello riescono effettivamente a ingannarlo. In maniera più formale,

l'ASR è definito come segue:

$$ASR = (\frac{N_{Successful}}{N_{Total}}) \times 100$$

dove $N_{Successful}$ è il numero di adversarial examples che hanno compromesso il modello e N_{Total} è il numero totale di adversarial examples generati. Un ASR elevato indica una maggiore vulnerabilità del modello agli attacchi.

Un'ulteriore metrica è rappresentata dall'*accuracy*. L'accuratezza quantifica la capacità del modello di mantenere prestazioni di classificazione corrette in presenza di input manipolati. Per determinare l'efficacia di un attacco, si effettua un confronto sistematico tra l'accuratezza del modello valutata su dati originali non perturbati e quella misurata su dati sottoposti a perturbazioni. Una degradazione significativa dell'accuratezza in presenza di input manipolati indica l'efficacia dell'attacco nel compromettere le prestazioni del classificatore.

In maniera complementare all'accuratezza, che misura i successi del modello, possiamo utilizzare l'*error rate* per misurarne i fallimenti. L'analisi degli errori confronta il tasso di errore del modello prima e dopo l'implementazione di schemi di difesa; di conseguenza, un tasso di errore inferiore dopo la difesa indica una difesa efficace. L'obiettivo ottimale consiste nella massimizzazione dell'accuratezza e nella contemporanea minimizzazione del tasso di errore. Queste due metriche sono intrinsecamente correlate, essendo l'una il complemento dell'altra, e pertanto il loro andamento risulta inversamente proporzionale durante il processo di ottimizzazione del modello.

Nell'ambito della valutazione dei metodi di adversarial training, che verranno approfonditi nei capitoli successivi, emergono diverse metriche specifiche per l'analisi dell'efficacia delle strategie di difesa. Una di queste è rappresentata dal numero medio di query effettuate dall'attaccante verso il modello: un elevato numero di interrogazioni necessarie per compromettere il sistema indica una maggiore robustezza del meccanismo di difesa. Un'altra misura è l'*embedding testing*, in cui metriche di similarità come l'*edit distance*, *Jaccard similarity coefficient* e *semantic similarity* vengono utilizzate per valutare l'utilità degli adversarial examples generati. Nello specifico:

1. Edit Distance: misura il numero di modifiche necessarie per trasformare un testo in un altro.

2. Jaccard Similarity Coefficient: valuta la similarità tra due insiemi di parole.

3. Semantic Similarity: determina quanto il significato del testo modificato rimane simile a quello originale.

L'analisi della loss function costituisce un ulteriore strumento di valutazione per i metodi di adversarial training, come la regolarizzazione e il virtual adversarial training. L'efficacia di queste tecniche è dimostrata dalla riduzione del tasso di errore e dal contenimento del fenomeno dell'overfitting.

Un aspetto fondamentale della valutazione rimane il giudizio umano: gli annotatori valutano la naturalezza degli esempi avversari confrontandoli con gli input originali. Questa analisi risulta cruciale per determinare la qualità degli adversarial examples impiegati nell'adversarial training, poiché la debolezza dell'attacco è direttamente proporzionale alla robustezza della strategia di difesa.

A complemento di queste metriche specifiche, è possibile integrare l'analisi con misure tradizionali di valutazione delle prestazioni, come *precision*, *recall* e *F1-score*, in base alle specifiche esigenze del contesto applicativo.

L'ecosistema degli strumenti di valutazione della sicurezza dei modelli comprende l'introduzione di framework specializzati, tra questi nel campo del Natural Language Processing spicca *TextAttack* (Morris et al., 2020). TextAttack offre un'infrastruttura completa per la progettazione, l'implementazione e la valutazione sistematica di attacchi avversari, fornendo al contempo un insieme completo di metriche e strumenti di analisi comparativa. La versatilità di questi framework consente ai ricercatori di condurre analisi approfondite sulla robustezza dei modelli in molteplici scenari, facilitando sia l'identificazione di vulnerabilità potenziali sia lo sviluppo di difese più efficaci.

2.4 Principali Sfide e Limitazioni

Nello scenario presentato, abbiamo visto come vi sia una grande eterogeneità, sia per quanto riguarda le metodologie di attacco, sia per la valutazione di queste. Per questi motivi, ogni caso va valutato in maniera indipendente; questa è anche una delle ragioni per cui è così difficile sviluppare strategie difensive efficaci, come vedremo nel capitolo 3. Una delle questioni più critiche riguarda la tensione intrinseca tra la robustezza dei modelli e la loro capacità di generalizzazione. I modelli di machine learning sono progettati per estrarre pattern dai dati e generalizzare a nuovi esempi, ma questa stessa caratteristica li rende vulnerabili a perturbazioni mirate. Paradossalmente, modelli con maggiore capacità di generalizzazione possono risultare più suscettibili agli adversarial attack, suggerendo l'esistenza di un trade-off fondamentale tra robustezza e capacità di apprendimento. Questo dilemma è ben documentato da (Goodfellow et al., 2015) in *Explaining and Harnessing Adversarial Example*:

> *"An intriguing aspect of adversarial examples is that an example generated for one model is often misclassified by other models, even when they have different architecures or were trained on disjoint training sets. Moreover, when these different models misclassify an adversarial example, they often agree with each other on its class."*

La capacità degli adversarial example di rimanere efficaci attraverso modelli diversi, solleva questioni profonde sulla natura delle vulnerabilità nei sistemi di machine learning. Questo fenomeno non si manifesta solo tra modelli diversi ma anche tra task differenti, evidenziando come le perturbazioni possano sfruttare caratteristiche fondamentali del processo di apprendimento. Citando nuovamente la medesima ricerca, Goodfellow et al. (2015) ha provato a dare una spiegazione al fenomeno:

> *"To explain why mutiple classifiers assign the same class to adversarial examples, we hypothesize that neural networks trained with current methodologies all resemble the linear classifier learned on the same training set. This reference classifier is able to learn approximately the same classification weights when trained on different subsets of the training set, simply because machine learning algorithms are able to generalize. The stability of the underlying classification weights in turn results in the stability of adversarial examples."*

Nonostante i numerosi approcci proposti per la difesa contro gli adversarial attack, non esiste ancora una soluzione definitiva che garantisca la robustezza mantenendo prestazioni competitive sui task originali. I metodi di difesa spesso soffrono di limitazioni significative: possono essere computazionalmente costosi, ridurre le performance del modello su input non perturbati, o risultare efficaci solo contro specifiche tipologie di attacco.

I motivi che possono portare i modelli ad essere così vulnerabili agli adversarial attack sono molteplici, e in gran parte derivano dalla natura approssimativa di questi. Nonostante i numerosi studi, i ricercatori non hanno ancora sviluppato una teoria unificata che spieghi completamente perché i modelli di machine learning siano così suscettibili a piccole perturbazioni dell'input. Questa lacuna teorica limita la nostra capacità di sviluppare soluzioni definitive al problema e rende necessario approfondire la ricerca su questi temi. Paragonare i modelli di intelligenza artificiale a sfere di cristallo è deliberatamente provocatorio e potenzialmente fuorviante, tuttavia questa metafora illustra efficacemente il duplice aspetto di questi strumenti: la loro potenza da un lato e la loro intrinseca fragilità dall'altro.

3 Adversarial Attack in NLP

Prima di addentrarci nel merito del tema del capitolo, forniamo un'introduzione allo scenario del Natural Language Processing. Il NLP rappresenta uno dei settori più dinamici e in rapida evoluzione nell'ambito dell'Intelligenza Artificiale. Questa disciplina si pone l'obiettivo di sviluppare sistemi in grado di comprendere, interpretare e generare linguaggio naturale umano. Negli ultimi anni, grazie anche all'avvento dei modelli neurali profondi e in particolare dei *transformer*, il campo ha registrato progressi straordinari, portando alla creazione di sistemi sempre più sofisticati e capaci di gestire task linguistici complessi. Gli scenari applicativi del NLP sono estremamente vasti e

permeano ormai numerosi aspetti della vita quotidiana. Solo alcuni di questi sono, ad esempio, i sistemi di traduzione automatica, che hanno reso accessibili contenuti in lingue diverse a un pubblico globale. I sistemi di comprensione e generazione del testo trovano applicazione nei chatbot e negli assistenti virtuali, che sono diventati strumenti fondamentali per il customer service e l'automazione dei processi aziendali. Modelli di *sentiment analysis* supportano processi decisionali in ambito finanziario e di business intelligence, mentre i sistemi di *summarization* aiutano a gestire la crescente mole di informazioni digitali. Un ruolo particolarmente critico è quello dei sistemi di content moderation, che utilizzano tecniche di NLP per identificare e filtrare contenuti inappropriati o dannosi sulle piattaforme social. Parallelamente, i modelli di *question answering* sono diventati componenti essenziali, migliorando significativamente l'accesso alle informazioni. Nel contesto aziendale, i sistemi di analisi dei documenti automatizzati stanno trasformando settori come quello legale e amministrativo, permettendo l'estrazione efficiente di informazioni da grandi volumi di documenti. Tuttavia, proprio la pervasività e la criticità di queste applicazioni rendono particolarmente rilevante il tema della sicurezza e della robustezza dei sistemi di NLP. La possibilità di manipolare questi sistemi attraverso adversarial attack solleva preoccupazioni significative, specialmente considerando il ruolo sempre più centrale che questi ricoprono in processi decisionali automatizzati e nella gestione di informazioni sensibili.

3.1 Problemi nel Contesto NLP

Dopo aver fornito le premesse necessarie, è opportuno analizzare le sfide principali che caratterizzano questo ambito di ricerca. Il Natural Language Processing (NLP) si distingue significativamente da altri settori dell'intelligenza artificiale, in particolare dalla computer vision, per la natura dei dati trattati. Mentre quest'ultima disciplina si occupa principalmente dell'analisi di dati visivi sotto forma di immagini, il NLP opera su dati testuali, una caratteristica che introduce specifiche complessità e richiede approcci metodologici distintivi.

La ricerca di (Zhang et al., 2019), *Adversarial Attacks on Deep Learning Models in Natural Language Processing: A Survey*, affronta la tematica delle differenze in maniera estremamente chiara. Sottolinea come le stesse metodologie di attacco adottate per le immagini non possano essere utilizzate anche per il testo per via di tre differenze chiave. La prima riguarda il dominio su cui operano i dati; contrariamente alle immagini che sono di tipo continuo, i dati testuali sono simbolici, quindi di natura discreta. Questo comporta una differente rappresentazione dei dati e pertanto anche un diverso trattamento. Nonostante questo esistono delle tecniche di *encoding* ed *embedding* che associano un vettore a ciascun input, rendendo possibile la rappresentazione delle parole nello spazio.

Un'altra importante distinzione riguarda la percezione delle perturbazioni negli adversarial example. A differenza delle immagini, in cui difficilmente un essere umano percepisce modifiche sui pixel, cambiamenti nei testi come la modifica di caratteri o parole possono risultare evidenti. Questo comporta un grado di complessità degli attacchi non indifferente nel caso testuale; inoltre, le eventuali modifiche potrebbero anche essere corrette o identificate da sistemi di controllo grammaticale automatici.

La terza differenza chiave riguarda la semantica, quindi il significato. Nel caso visivo le perturbazioni minime non cambiano la semantica di un'immagine, in quanto la modifica dei pixel, ad esempio, non trasforma un animale in un altro. Nel caso testuale invece, perturbazioni di parole o frasi, cambiano facilmente il significato del testo, rendendo le modifiche evidenti. Ricordiamo infatti che l'obiettivo degli adversarial attack è quello di generare perturbazioni impercettibili dall'uomo ma in grado di ingannare i modelli, quindi la modifica della semantica dell'input è contraria all'obiettivo.

Gli adversarial attack applicati al testo devono quindi, soddisfare tre requisiti essenziali: la conservazione del significato originale, l'implementazione di perturbazioni non rilevabili dall'essere umano e una rappresentazione efficiente dei dati nello spazio.

3.1.1 Rappresentazioni Vettoriali

Come menzionato in precedenza, esistono diverse strategie per ottenere una rappresentazione vettoriale delle parole. Questi metodi possono essere classificati in tre gruppi: *word-count based encoding*, *one-hot encoding*, e *dense encoding*. Alla prima categoria appartengono tecniche come *bag-of-words* (BoW), che propongono una rappresentazione non strutturata di tutte le parole di un documento testuale, definita esclusivamente sulla base della frequenza, trascurando l'ordine e il

contesto delle parole. In altri termini, ciascun documento è descritto mediante un vettore la cui lunghezza corrisponde al vocabolario, dove ogni posizione è associata a una parola e il valore rappresenta il numero di occorrenze di quella parola nel documento. Un'altra tecnica degna di nota è *term frequency - inverse document frequency* (TF-IDF), che incrementa l'efficacia del BoW ponderando le parole in funzione della loro rilevanza nei documenti. Il principio fondamentale è che alcune parole semanticamente irrilevanti (ad esempio, il, alcuni, ecc.) possono presentare una frequenza di termini elevata e, di conseguenza, un peso maggiore in un modello. La TF-IDF si propone di correggere questo aspetto riducendo il peso attribuito a tali parole.

La seconda categoria riguarda il caso delle codifiche one-hot. In questo genere di codifica si assume che ogni parola sia rappresentata in vettori di dimensione V, dove V rappresenta la dimensione del vocabolario. I vettori generati sono binari, in quanto a ciascuna componente è associato il valore 0, mentre il valore 1 (*hot*) è associato alla componente che rappresenta la parola corrente. Analogamente alle tecniche appartenenti alla categoria precedente, anche la codifica one-hot presenta le stesse limitazioni. Una criticità fondamentale risiede nella mancata considerazione del contesto linguistico, poiché questa metodologia non preserva l'ordine sequenziale delle parole all'interno del testo. Un'ulteriore limitazione è rappresentata dall'incapacità di catturare le relazioni semantiche tra i termini: parole con significati affini vengono infatti rappresentate da vettori differenti tra loro, perdendo così qualsiasi informazione sulla loro similarità concettuale.

La terza categoria comprende le tecniche di codifica densa, dove le parole vengono rappresentate come vettori numerici in uno spazio multidimensionale, comunemente denominati *word embedding*. Questo approccio si fonda su due principi fondamentali: l'ipotesi distribuzionale, secondo cui parole semanticamente simili occupano posizioni prossime nello spazio vettoriale, e la semantica distribuzionale, secondo cui la derivazione del significato di una parola può essere dedotta dal contesto in cui essa appare. Attraverso questi principi teorici, la rappresentazione vettoriale permette di catturare sia le relazioni semantiche che quelle sintattiche tra le parole, basandosi sui pattern contestuali presenti nei dati di addestramento. Tuttavia, gli embedding statici, come quelli generati dagli algoritmi *Word2Vec* e *GloVe*, presentano alcune limitazioni intrinseche. La principale criticità risiede nella natura statica di queste rappresentazioni: ogni termine del vocabolario viene associato a un unico vettore fisso nello spazio degli embedding, indipendentemente dal contesto in cui occorre. Inoltre, questi modelli si limitano all'analisi del contesto locale immediato, senza considerare la struttura sintattica complessiva del testo.

Per sopperire a queste limitazioni, i recenti sviluppi hanno portato alla creazione di tecniche come *contextural embeddings* e *transformer-based embedding*, in grado di produrre rappresentazioni che cambiano a seconda della frase in cui la parola appare. In particolare, un *Transformer* è un modello basato su un'architettura neurale con un meccanismo chiave chiamato *self-attention*. Questo gli permette di analizzare tutte le parole di una frase contemporaneamente e capire quali parole sono più importanti tra loro.

3.1.2 Misura delle Perturbazioni

Ora che abbiamo fornito delle soluzioni per rappresentare il testo nello spazio, è necessario poter misurare le perturbazioni, cioè la distanza tra il dato originale e l'adversarial example. Tuttavia, come abbiamo già più volte sottolineato, nella misurazione è fondamentale considerare anche la correttezza sintattica e semantica del testo.

Proprio per l'impossibilità di poter generare perturbazioni del tutto indistinguibili dai dati originali, possiamo raggruppare gli attacchi in due gruppi, in base alla nozione di similarità associata. L'immagine in figura 1 è tratta dalla documentazione del framework TextAttack (Morris et al., 2020), e mostra i risultati ottenuti da un attacco utilizzando due strategie di similarità differenti. In particolare, questi risultati sono stati ricavati attaccando un modello *LSTM* addestrato sul dataset di sentiment analysis di *Rotten Tomatoes Movie Review*. Si tratta di esempi avversari reali, generati utilizzando gli attacchi *DeepWordBug* e *TextFooler* messi a disposizione da TextAttack.

La prima categoria di similarità riguarda la *visual similarity*, che comprende metodologie di attacco mirate alla manipolazione minima dei caratteri al fine di alterare la predizione del modello. Alcune varianti di questi attacchi si concentrano sull'introduzione di errori ortografici che emulano le comuni imperfezioni nella digitazione umana. Tuttavia, questa tipologia di attacchi presenta una significativa vulnerabilità: l'implementazione di sistemi automatici di correzione ortografica e gram-

Original Input	Connoisseurs of Chinese film will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: **Positive (77%)**
Adversarial example [Visually similar]	**Aonnoisseurs** of Chinese film will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: **Negative (52%)**
Adversarial example [Semantically similar]	Connoisseurs of Chinese **footage** will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: **Negative (54%)**

Figure 1: Due differenti tipologie di adversarial example in NLP.

maticale può efficacemente neutralizzare tali tentativi di manipolazione, rendendo questa strategia sostanzialmente inefficace.

La seconda categoria è rappresentata dalla *semantic similarity*, dove un adversarial example viene considerato valido solamente se mantiene una sostanziale equivalenza semantica con l'input originale. In questo contesto, le strategie predominanti si basano sulla sostituzione sistematica mediante sinonimi o sulla riformulazione attraverso parafrasi, con il vincolo fondamentale che l'input originale e quello perturbato debbano generare predizioni differenti da parte del modello.

Le misure utilizzate per valutare le perturbazioni testuali negli attacchi avversari possono essere suddivise in diverse categorie, alcune delle quali già citate, a seconda dell'aspetto del testo che si intende preservare.

Per garantire che gli esempi avversari risultino plausibili e non facilmente individuabili, vengono impiegati strumenti di verifica grammaticale e sintattica. In questo contesto la *perplexity* viene spesso utilizzata per misurare la qualità linguistica del testo generato, mentre il controllo sulla validità delle parafrasi assicura che le modifiche apportate non alterino il significato originale. La perplexity quantifica quanto bene un modello predice una sequenza di parole, è una misura della "confusione" del modello riguardo ai dati che deve processare. Più è bassa, meglio il modello riesce a fare previsioni.

Un aspetto cruciale nella valutazione delle perturbazioni testuali è la preservazione del significato. Questo viene generalmente misurato confrontando i vettori di parole attraverso metriche di similarità o distanza, come la distanza euclidea o la similarità coseno. Questi strumenti permettono di verificare che il testo modificato mantenga una relazione semantica con l'originale.

Parallelamente, esistono metriche che si concentrano sulle modifiche apportate al testo. L'edit distance misura il numero minimo di operazioni necessarie per trasformare un testo in un altro, utilizzando tecniche come la distanza di *Levenshtein*, che considera inserzioni, rimozioni e sostituzioni, o il *Word Mover's Distance* (WMD), che opera direttamente sugli embedding delle parole per quantificare la distanza tra due documenti.

Infine, la similarità tra testi può essere valutata anche attraverso approcci basati sugli insiemi, come il *Jaccard Similarity Coefficient*, che misura il grado di sovrapposizione tra due documenti confrontando il numero di parole in comune rispetto al totale. Queste misure, nel loro insieme, forniscono strumenti efficaci per valutare la validità e l'efficacia delle perturbazioni testuali negli adversarial attack, bilanciando la correttezza grammaticale, la preservazione semantica e il grado di modifiche apportate al testo.

3.2 Tecniche e Strategie di Attacco in NLP

Prima di procedere con un'analisi approfondita delle strategie di attacco, è opportuno fornire una panoramica delle caratteristiche e delle classificazioni che possono emergere in base al livello di dettaglio dell'analisi testuale. Questa contestualizzazione preliminare permetterà di comprendere meglio le diverse sfaccettature che caratterizzano gli attacchi in base alla natura intrinseca del testo.

3.2.1 GRANULARITÀ

Riprendendo quanto già accennato nel capitolo precedente, gli attacchi nel contesto del Natural Language Processing possono essere catalogati in base al livello di dettaglio delle perturbazioni. La granularità è ben descritta da (Goyal et al., 2023) in *Survey of adversarial defences and robustness in NLP*. Nello specifico il grado di modifica può essere suddiviso in *character level*, *word level*, *sentence level* e *multi-level* adversarial attack.

La prima categoria riguarda gli attacchi a livello di carattere, questi modificano le sequenze di input manipolando singoli caratteri attraverso inserimenti, cancellazioni e scambi. Sebbene efficaci, questi attacchi possono essere facilmente rilevati da correttori ortografici. Le tecniche includono l'aggiunta di rumore naturale (errori ortografici reali) e sintetico (scambio o randomizzazione di caratteri, sostituzione con caratteri adiacenti sulla tastiera, inserimento di segni di punteggiatura, modifica degli spazi).

La seconda tipologia comprende gli attacchi a livello di parola, che operano modificando parole intere anziché singoli caratteri, applicando tecniche di inserimento, eliminazione e sostituzione. Questi si distinguono in tre principali approcci metodologici. Il primo approccio riguarda i metodi basati sul gradiente, che monitorano le variazioni del gradiente per ogni perturbazione dell'input, considerando efficace una modifica quando altera la probabilità di classificazione. Successivamente troviamo le tecniche basate sull'importanza, che partono dal presupposto che le parole con punteggi di attenzione molto alti o molto bassi siano fondamentali nelle previsioni dei modelli di *self-attention*. Queste parole vengono sistematicamente perturbate fino al successo dell'attacco. Infine, i metodi basati sulla sostituzione, che rimpiazzano casualmente le parole con alternative semanticamente e sintatticamente affini, utilizzando rappresentazioni vettoriali dense.

La terza categoria fa riferimento agli attacchi a livello di frase, questi rappresentano una forma più sofisticata di manipolazione testuale, operando su gruppi di parole piuttosto che su elementi singoli. Questa tipologia offre maggiore flessibilità, poiché una frase alterata può essere inserita in qualsiasi punto del testo, purché rimanga grammaticalmente corretta. Particolarmente efficaci in task come il Natural Language Inference, i sistemi di question-answering, la traduzione automatica, il Natural Language Understanding e la text classification. Alcune strategie di attacco sentence level sono progettate per preservare l'etichetta originale dell'input, venendo semplicemente concatenate al testo principale. In questi casi, il comportamento corretto del modello sarebbe mantenere l'output originale, mentre l'attacco ha successo se provoca un cambiamento nell'output o nell'etichetta assegnata.

Infine, i multi-level adversarial attack rappresentano un approccio complesso che integra varie tecniche precedentemente discusse, al fine di creare perturbazioni meno percettibili dagli esseri umani e garantire tassi di successo più elevati.

3.2.2 TIPOLOGIE DI ATTACCO

Nel capitolo 2.2 abbiamo visto come gli adversarial attack possano essere raggruppati in diverse categorie in base alla metodologia di attacco e alle sue implicazioni pratiche. La figura 2 ci aiuta a consolidare il panorama degli attacchi avversari e riassume in parte quanto già detto.

Esiste una distinzione intrinseca tra attacchi volti a minacciare modelli neurali e non neurali, basata ovviamente sulla struttura stessa dei modelli. Tuttavia possiamo distinguere in maniera più semplice gli attacchi in base alla conoscenza, in quanto tipologie di attacco black-box possono essere applicate sia nel caso di modelli neurali sia non neurali, tenendo presente che l'efficacia dell'attacco può variare.

In questa parte forniremo una panoramica delle varie tipologie di attacco tenendo presente la distinzione tra white-box e black-box attack. Nel farlo riprenderemo quanto già detto da (Zhang et al., 2019) in *Adversarial Attacks on Deep Learning Models in Natural Language Processing: A Survey* e (Morris et al., 2020) in *Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP*.

White-Box Attack

Nello scenario white-box, l'attaccante dispone di accesso completo alle informazioni critiche del modello target, inclusi i parametri, l'architettura e la funzione di perdita (loss function). La disponibilità di questi dettagli tecnici consente di sviluppare attacchi particolarmente sofisticati ed efficaci

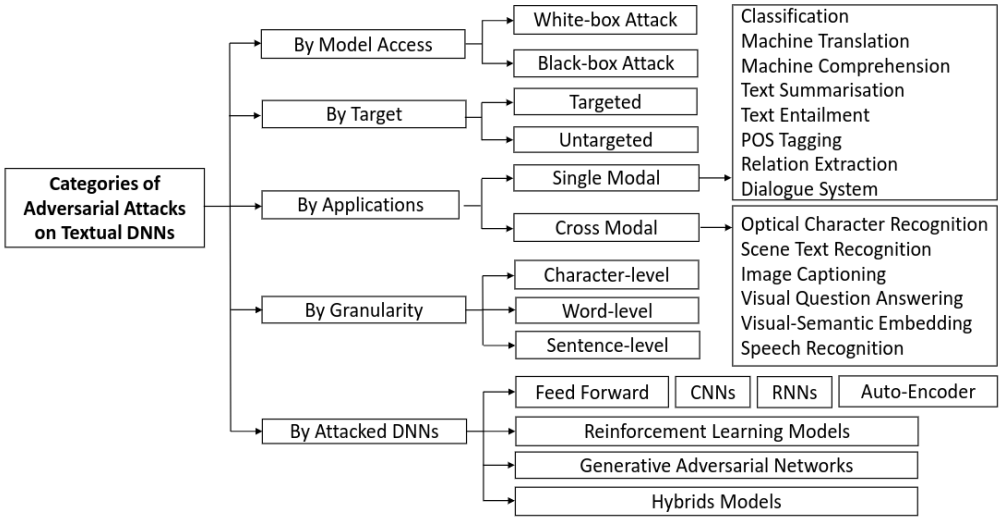

Figure 2: Categorie di adversarial attack in NLP

contro il sistema. È tuttavia fondamentale evidenziare come, in contesti reali, l'accesso a informazioni così dettagliate rappresenti uno scenario piuttosto raro, poiché nella maggior parte delle implementazioni pratiche questi elementi rimangono riservati e adeguatamente protetti.

FGSM. L'idea chiave di FGSM è perturbare l'input nella direzione che massimizza la perdita rispetto all'etichetta corretta. Muovendosi nella direzione del gradiente della loss, si massimizza, infatti, la probabilità di errore del modello. Fast Gradient Sign Method rappresenta uno dei primi metodi di attacco sviluppati per le immagini, che ha successivamente ispirato diverse strategie nel campo dell'elaborazione del testo, basate su inserimento, modifica e rimozione di elementi testuali. La strategia di inserimento prevede l'aggiunta di frasi specificamente selezionate per influenzare la classificazione verso una classe target. La modifica si concentra sulla sostituzione di caratteri con errori di ortografia comuni o caratteri visivamente simili. La strategia di rimozione, invece, elimina elementi non essenziali come aggettivi o avverbi che contribuiscono alla classificazione originale. Sviluppi successivi hanno perfezionato questo approccio, introducendo strategie più sofisticate che combinano rimozione, aggiunta e sostituzione di parole, prestando particolare attenzione alla preservazione della grammatica e della struttura sintattica del testo originale.

HotFlip. HotFlip rappresenta un approccio nella generazione di esempi avversari, basato su operazioni atomiche a livello di carattere. Il metodo si concentra su tre operazioni fondamentali: scambio, inserimento ed eliminazione di caratteri. Il sistema valuta l'impatto di queste modifiche attraverso l'uso di derivate direzionali, permettendo di identificare le alterazioni più efficaci per influenzare l'output del modello. Successivi sviluppi hanno esteso le capacità di HotFlip introducendo attacchi mirati più sofisticati. Questi includono attacchi controllati, che mirano a rimuovere parole specifiche dall'output, e attacchi targettizzati, che sostituiscono parole specifiche con altre predefinite. Per preservare la leggibilità e la coerenza del testo, questi metodi includono limitazioni sul numero massimo di modifiche consentite.

JSMA. La Jacobian Saliency Map Adversary si distingue da FGSM per il suo approccio basato sulle derivate invece che sui gradienti. Come altri attacchi di questa categoria è inizialmente stato sviluppato per generare perturbazioni mirate verso le immagini, tuttavia è stato successivamente adattato per poter funzionare anche in contesti basati su testo. Questo metodo valuta la sensibilità dell'output del modello neurale rispetto a ciascuna componente dell'input attraverso l'uso della matrice Jacobiana [1]. Le Jacobian Saliency Map classificano il contributo di ogni componente dell'input rispetto all'obiettivo dell'attacco, permettendo una selezione mirata delle perturbazioni. Questo approccio

[1]Data una funzione che produce un vettore di output a partire da un vettore di input, la matrice Jacobiana descrive come ogni elemento dell'output cambia a variare di ogni singolo elemento dell'input.

offre agli attaccanti un controllo più granulare sulle perturbazioni, consentendo una maggiore precisione nella generazione degli esempi avversari rispetto ai metodi basati sul gradiente.

Attention-based. Gli attention-based attack sono attacchi avversari che sfruttano i meccanismi di *self-attention* [2] nei moderni modelli basati su *Transformer*, per identificare e modificare parti del testo con maggiore impatto sulla previsione del modello. L'attacco consiste nello scambiare le parole chiave in queste frasi con altre parole scelte da un vocabolario noto o rimuovendo intere frasi considerate rilevanti. Esistono delle varianti che si distinguono principalmente per il livello di dettaglio con cui vengono applicate le perturbazioni.

Attacco	Granularità	Target	Perturbazione
FGSM	Carattere, Parola	Y, N	Gradiente
HotFlip	Carattere	Y	Gradiente
JSMA	Carattere, Parola	Y	Jacobian Matrix
Attention-based	Parola, Frase	Y, N	Self-Attention

Table 1: Sintesi delle differenze degli attacchi white-box

La tabella 1 riassume le caratteristiche e le differenze principali degli attacchi white-box enunciati. Un aspetto particolarmente rilevante nell'ambito degli attacchi avversari riguarda la loro applicazione nei modelli di classificazione binaria. In questo specifico contesto, la tradizionale distinzione tra attacchi mirati (targeted) e non mirati (untargeted) perde di significato, poiché entrambe le tipologie di attacco producono il medesimo effetto. Inoltre, è fondamentale evidenziare come questo campo di studi sia caratterizzato da una notevole dinamicità e da un'evoluzione continua della ricerca. Questa rapida progressione comporta frequentemente lo sviluppo di nuove varianti e perfezionamenti delle strategie di attacco precedentemente descritte. Le metodologie presentate rappresentano quindi solo una selezione delle principali tecniche documentate in letteratura.

Black-Box Attack

Nello scenario black-box, in netto contrasto con gli attacchi white-box, l'attaccante opera in condizioni di accesso limitato alle informazioni del modello target, potendo osservare esclusivamente la relazione tra input forniti e output generati. Questo scenario rappresenta un contesto operativo significativamente più realistico e frequente nelle applicazioni pratiche, poiché riflette le effettive condizioni di accessibilità dei sistemi di apprendimento automatico nel mondo reale. Questa limitazione nelle informazioni disponibili ha portato allo sviluppo di strategie di attacco specificamente progettate per operare con successo in tali condizioni. Le metodologie che verranno presentate sono state concepite e ottimizzate per sfruttare efficacemente questa base di conoscenza ridotta, dimostrando come sia possibile compromettere la sicurezza di un sistema anche in assenza di informazioni dettagliate sulla sua struttura interna.

Concatenation Adversaries. Questa tecnica si basa sull'aggiunta di frasi di disturbo, prive di significato rilevante, alla fine del paragrafo target; un esempio è proposto dalla figura 3.

Queste frasi, pur non alterando la semantica del testo originale o le risposte alle domande, riescono a indurre in errore il modello. Le frasi di disturbo possono essere generate seguendo due approcci principali: attraverso la creazione accurata di frasi informative ma fuorvianti, oppure mediante la costruzione di sequenze arbitrarie di parole comuni selezionate da un pool predefinito. La generazione di queste perturbazioni avviene attraverso un processo iterativo, descritto dalla figura 4, continuando fino a quando non si ottiene il cambiamento desiderato nell'output.

Edit Adversaries. Questa tipologia di attacchi mira a compromettere i modelli attraverso modifiche mirate ai dati di input. Invece di manipolare direttamente i parametri interni del modello, gli edit adversaries agiscono a livello superficiale, alterando il testo in modi che appaiono naturali e impercettibili agli occhi umani, ma che sono sufficienti a confondere il modello, come mostrato in figura 5.

Questi attacchi si concentrano sulla perturbazione dei dati di input su diversi gradi di granularità, con strategie di modifica che includono sostituzioni, eliminazioni, aggiunte e scambi. La figura 6

[2] meccanismo che permette di valutare l'importanza di ciascuna parola nel testo e assegnare loro un peso in base alla rilevanza per il contesto.

Article: Super Bowl 50
Paragraph: "*Peyton Manning became the first quarterback ever to lead two different teams to multiple Super Bowls. He is also the oldest quarterback ever to play in a Super Bowl at age 39. The past record was held by John Elway, who led the Broncos to victory in Super Bowl XXXIII at age 38 and is currently Denver's Executive Vice President of Football Operations and General Manager. Quarterback Jeff Dean had jersey number 37 in Champ Bowl XXXIV.*"
Question: "*What is the name of the quarterback who was 38 in Super Bowl XXXIII?*"
Original Prediction: John Elway
Prediction under adversary: Jeff Dean

Figure 3: Applicazione pratica di Concatenation Adversaries

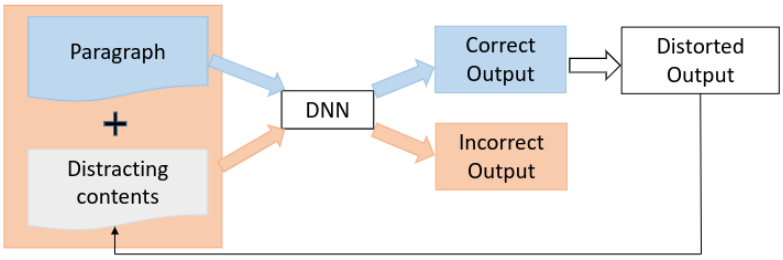

Figure 4: Schema di funzionamento generale degli attacchi basati su concatenazione

chiarisce il funzionamento della strategia. L'attaccante cerca di individuare quali parole o caratteri del testo hanno un impatto maggiore sulla decisione del modello.

Nella modalità più tradizionale, il sistema viene interrogato tramite query e si osserva come varia la confidenza nelle predizioni quando viene modificato un token. Alcuni degli attacchi principali di questa categoria sono *TextFooler*, *TextBugger* e *DeepWordBug*, i quali si distinguono per il livello di dettaglio su cui operano e per il modo in cui valutano l'importanza del testo.

Paraphrase-based Adversaries. Questi attacchi rappresentano una strategia complessa, in cui l'obiettivo è ingannare il modello attraverso la creazione di variazioni del testo originale che ne mantengono intatto il significato. A differenza di altri tipi di attacchi che si concentrano su modifiche superficiali come errori di battitura o sostituzioni di singole parole, gli attacchi Paraphrase-based mirano a generare parafasi del testo originale, ovvero riformulazioni che esprimono lo stesso contenuto in modo diverso; un'illustrazione del funzionamento è fornita dalla figura 7. Nelle versioni più sofisticate la parafrasi può essere generata attraverso l'utilizzo di modelli encoder-decoder, per poi essere valutata e confrontata con la frase originale mediante l'uso di metriche di similarità o modelli terzi. Nonostante la grande complessità, la difesa da questo genere di attacchi risulta particolarmente difficile, in quanto essi risultano difficili da rilevare.

GAN-based Adversaries. Gli attacchi avversari basati su GAN (Generative Adversarial Network) rappresentano un approccio innovativo finalizzato alla generazione di esempi avversari più naturali. Le GAN sono composte da due reti, un generatore e un discriminatore. Il discriminatore deve distinguere tra campioni reali e generati, mentre il generatore crea campioni realistici che hanno lo scopo di ingannare il discriminatore. L'obiettivo di questi attacchi è quello di creare adversarial example che siano semanticamente equivalenti all'input originale ma che, grazie a variazioni sottili nella forma, portino il modello a produrre una risposta errata. Inoltre, il discriminatore, in alcuni casi può anche essere integrato nel processo di addestramento del modello target per migliorarne la robustezza. Questa categoria di attacchi rappresenta un approccio sofisticato nella generazione delle perturbazioni; tuttavia, l'impiego di reti avversarie comporta un notevole aumento dei costi.

Task: Sentiment Analysis. ***Classifier:*** Amazon AWS. ***Original label:*** 100% Negative. ***Adversarial label:*** 89% Positive.

Text: I watched this movie recently mainly because I am a Huge fan of Jodie Foster's. I saw this movie was made right between her 2 Oscar award winning performances, so my expectations were fairly high. ~~Unfortunately~~ **Unf0rtunately**, I thought the movie was ~~terrible~~ **terrib1e** and I'm still left wondering how she was ever persuaded to make this movie. The script is really ~~weak~~ **wea k**.

Figure 5: Applicazione pratica di edit adversaries

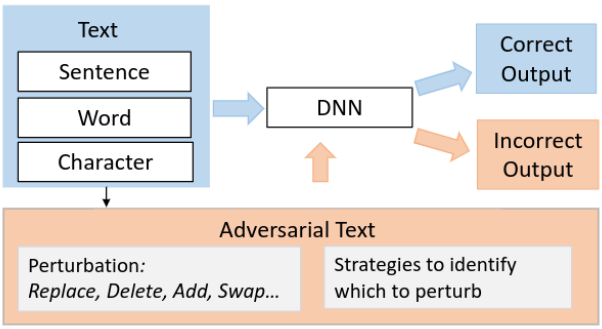

Figure 6: Schema di funzionamento generale degli attacchi basati su edit adversaries

Substitution. Gli attacchi basati su sostituzione rappresentano una categoria distintiva che trova applicazione sia nel contesto degli attacchi black-box che white-box. Questa metodologia si basa sull'addestramento di un modello surrogato che approssima i confini decisionali del modello target attraverso un processo di interrogazione sistematica, dal quale si ottengono le etichette necessarie per l'addestramento. Il risultato è la generazione di un nuovo modello che replica idealmente il comportamento del modello target, con il vantaggio significativo di avere pieno accesso ai suoi parametri interni. Questa caratteristica consente l'applicazione di strategie di attacco white-box sul modello surrogato, superando così le limitazioni tipiche degli scenari black-box e permettendo l'utilizzo di tecniche più sofisticate di generazione di esempi avversari.

Attacco	Granularità	Target	Perturbazione
Concatenation Adversaries	Carattere, Parola, Frase	N	elementi disturbanti
Edit Adversaries	Carattere, Parola, Frase	N	modifiche, eliminazioni
Paraphrase-based	Parola	N	parafrasi
GAN-based	Parola	N	reti avversarie
Substitution	-	Y, N	modelli surrogati

Table 2: Sintesi delle differenze degli attacchi black-box

La tabella 2 rappresenta una sintesi degli attacchi black-box analizzati. È doveroso puntualizzare ancora una volta come questi siano solo una parte degli attacchi analizzati e discussi in letteratura. Inoltre, ciascuna delle categorie menzionate in precedenza può manifestarsi in un numero considerevole di varianti, alcune di queste delle quali sono sintetizzate nella tabella Tabella 2.

Multi-Modal Attack

Un'importante considerazione nell'ambito degli adversarial attack riguarda i modelli multi-modali, che rappresentano uno degli sviluppi più significativi nel campo del Natural Language Processing. Questi modelli avanzati sono caratterizzati dalla capacità di elaborare e integrare simultaneamente informazioni provenienti da diverse modalità, tra cui testo, immagini, audio e video. Tale caratteristica consente loro di comprendere e generare contenuti con una ricchezza semantica superiore rispetto ai modelli tradizionali. Le applicazioni di questi sistemi spaziano dalla conversione speech-to-text alla traduzione image-to-text, includendo anche le rispettive trasformazioni inverse. Nel contesto degli adversarial attack, sebbene sia possibile applicare alcune delle strategie di attacco

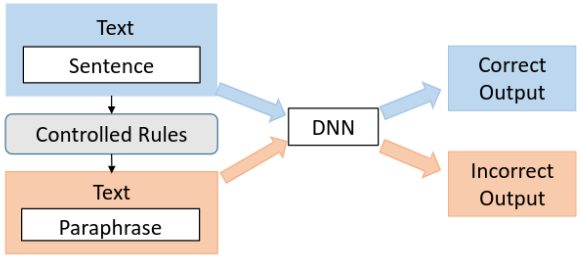

Figure 7: Schema di funzionamento generale degli attacchi basati su parafrasi

già consolidate, la natura multi-modale introduce ulteriori complessità. In particolare, si rende necessaria l'implementazione di opportune conversioni tra le diverse modalità per garantire l'efficacia degli attacchi, aumentando significativamente la complessità computazionale dell'intero processo.

3.3 Tecniche e Strategie di Difesa in NLP

Dopo aver esaminato approfonditamente le diverse strategie di attacco, le loro implicazioni e i relativi contesti applicativi, sorge spontanea la domanda: come è possibile, e se effettivamente realizzabile, costruire o rendere i modelli più resilienti a questo genere di minacce? In questa sezione, esamineremo le principali strategie difensive documentate in letteratura. Sebbene queste tecniche condividano l'obiettivo comune di potenziare la robustezza dei modelli contro gli adversarial attack, si distinguono significativamente per i loro principi operativi, finalità specifiche, metodologie di approccio e livelli di complessità implementativa.

Secondo quanto riportato da (Goyal et al., 2023) in *Survey of adversarial defences and robustness in NLP*, le principali strategie di difesa rientrano in tre categorie distinte: *Adversarial Training based*, *Perturbation Control based* e *Robustness by Certification*. La figura 8 ci fornisce uno schema rappresentativo dei principali metodi difensivi esistenti, parte dei quali verranno discussi in seguito.

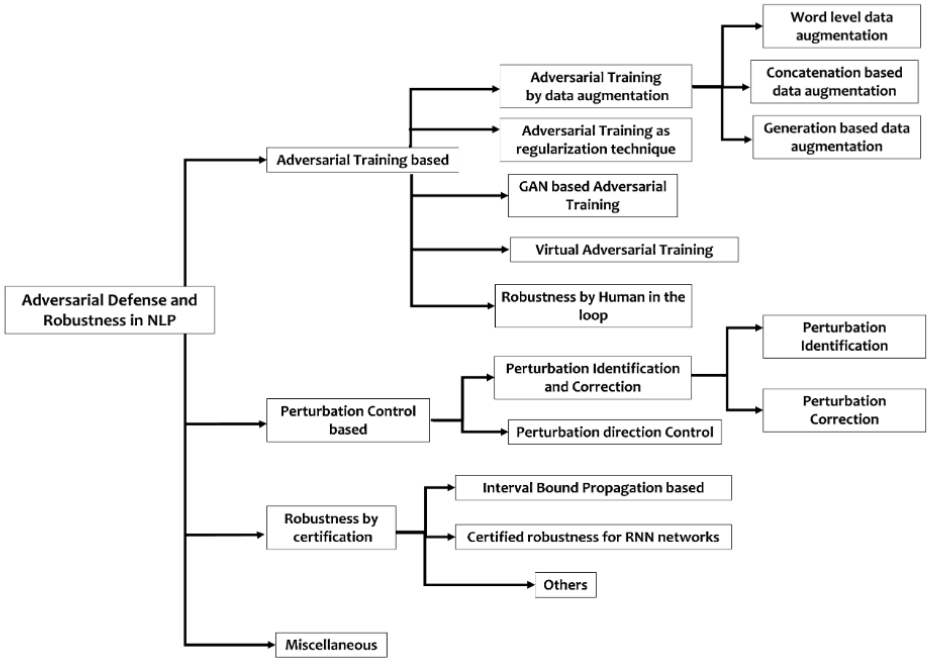

Figure 8: Panoramica sulle principali strategie di difesa in NLP

Inoltre discuteremo anche di un'altra strategia difensiva molto discussa in letteratura e analizzata in diverse ricerche come *Adversarial Attacks on Deep Learning Models in Natural Language Processing* di (Zhang et al., 2019), ovvero la *Defensive Distillation*.

3.3.1 ADVERSARIAL TRAINING

L'*adversarial training* rappresenta una metodologia di difesa sofisticata contro gli adversarial attack, basata sull'addestramento di modelli di machine learning attraverso l'utilizzo combinato di esempi standard ed avversari. L'obiettivo principale di questa tecnica è il potenziamento della robustezza del modello contro gli attacchi avversari, ottimizzando la sua capacità di generalizzazione e riducendo la sua vulnerabilità a input imprevisti.

L'implementazione dell'adversarial training può avvenire attraverso diverse metodologie. La data augmentation prevede la generazione di esempi avversari mediante tecniche di manipolazione del testo di input, come modifiche, inserimenti, eliminazioni o sostituzioni di parole o caratteri, che vengono successivamente integrati nel dataset di training originale per ampliare la varietà dei dati e preparare il modello a potenziali input malevoli. Le già citate Generative Adversarial Networks (GAN), possono essere applicate anche in contesti difensivi, dove un sistema generativo viene addestrato a produrre esempi avversari mentre un discriminatore apprende a distinguere tra esempi reali e artificiali, creando un processo che rafforza la robustezza del modello principale. Oltre a questi, l'approccio Human-In-The-Loop (HITL) integra l'expertise umana nel processo, coinvolgendo annotatori nella creazione di esempi avversari o nella valutazione e correzione degli output del modello.

L'efficacia dell'adversarial training è strettamente correlata alla qualità degli esempi avversari utilizzati: perturbazioni troppo deboli potrebbero risultare insufficienti per il miglioramento della robustezza, mentre esempi eccessivamente aggressivi rischiano di compromettere l'accuratezza su dati standard.

Tuttavia, l'adversarial training presenta alcune criticità significative. La specificità rappresenta un limite importante, poiché i modelli tendono a sviluppare robustezza principalmente contro le tipologie di attacchi utilizzate durante l'addestramento, mantenendo potenziali vulnerabilità verso nuove forme di attacco. Inoltre, l'elevato costo computazionale associato alla generazione di esempi avversari e al processo di addestramento può rendere questa tecnica difficilmente applicabile a modelli di grandi dimensioni o dataset estesi.

Nonostante queste sfide, l'adversarial training continua a rappresentare una delle strategie di difesa più efficaci nel campo del Natural Language Processing. La ricerca in questo ambito prosegue attivamente, con lo sviluppo di nuovi approcci volti a superare le limitazioni esistenti e a migliorare sia la robustezza che l'interpretabilità dei modelli NLP.

3.3.2 PERTURBATION CONTROL

La difesa basata su *perturbation control* rappresenta una strategia innovativa per contrastare gli attacchi avversari, focalizzandosi sul controllo delle perturbazioni nei dati di input anziché sulla generazione di esempi avversari o sulla modifica del modello stesso.

Questo approccio si sviluppa attraverso due componenti fondamentali: l'identificazione e la correzione delle perturbazioni. Nel processo di identificazione, il sistema si concentra sul riconoscimento e la localizzazione delle modifiche avversarie apportate all'input, impiegando metodologie avanzate come il monitoraggio delle variazioni nei gradienti, l'analisi delle caratteristiche linguistiche e l'utilizzo di modelli ausiliari per il rilevamento delle anomalie. La fase di correzione, successivamente, si occupa di ripristinare l'input alla sua forma originale o a una versione più affidabile, attraverso l'implementazione di *spell checker*, la sostituzione di sinonimi e l'applicazione di regole grammaticali per riparare strutture sintattiche compromesse.

Rispetto ad approcci tradizionali come l'adversarial training, questa strategia si distingue per non richiedere la generazione di esempi avversari durante l'addestramento, risultando così più efficiente dal punto di vista computazionale e particolarmente adatta in scenari dove la generazione di esempi avversari risulterebbe onerosa o complessa.

Tuttavia, è importante considerare che l'efficacia di questa strategia difensiva dipende fortemente dalla precisione nell'identificazione e correzione delle perturbazioni. Perturbazioni particolarmente complesse o ben mascherate potrebbero sfuggire al sistema di rilevamento, compromettendo l'efficacia della difesa. Pertanto, lo sviluppo continuo di tecniche sempre più sofisticate risulta fondamentale per mantenere l'efficacia del sistema contro attacchi avversari in costante evoluzione.

3.3.3 Robustness by Certification

La terza categoria riguarda la difesa robustness by certification, che rappresenta un approccio unico nella protezione dei modelli di machine learning contro gli attacchi avversari. Questa metodologia si distingue significativamente dalle tecniche come l'adversarial training o il perturbation control, poiché mira a prevenire gli attacchi, fornendo una garanzia formale sulla robustezza intrinseca del modello.

Il principio fondamentale di questo approccio risiede nell'addestramento del modello finalizzato a stabilire un limite superiore alla perdita nel caso peggiore delle perturbazioni, generando così un certificato di robustezza. Questo processo determina una regione di stabilità intorno a un dato input, all'interno della quale la classificazione del modello mantiene la sua coerenza, indipendentemente dalle perturbazioni introdotte, senza la necessità di esplorare esplicitamente lo spazio avversario.

Un tipico approccio di questa categoria è quello di aggiungere del rumore casuale all'input, eseguendo il modello più volte con differenti campioni di rumore. Se la maggior parte delle predizioni (sotto la distribuzione del rumore) indica la stessa classe, si può certificare che piccoli cambiamenti all'input non modificheranno la decisione.

Questa difesa offre una sicurezza matematica contro attacchi entro una certa ampiezza, rendendo il modello più affidabile in applicazioni critiche. Tuttavia, talvolta le certificazioni possono essere troppo conservative, limitando la capacità del modello di generalizzare in contesti reali dove gli attacchi possono essere più sofisticati. Inoltre, la definizione della regione sicura è generalmente un processo estremamente costoso dal punto di vista computazionale, limitandone l'applicabilità in contesti reali.

3.3.4 Distillation

L'ultima difesa che analizziamo è basata sulla distillation, nasce dall'idea di ridurre la sensibilità del modello agli attacchi avversari attraverso un processo di apprendimento che rende le sue decisioni meno brusche e i confini decisionali più morbidi. In pratica, si addestra inizialmente un modello *insegnante* sull'attività desiderata utilizzando i dati di training standard. Successivamente, un secondo modello, detto *studente*, viene addestrato a replicare le uscite del modello insegnante, in cui invece di utilizzare le etichette hard (ad esempio, la classe corretta), si utilizzano le probabilità di classe prodotte dal modello insegnante come target per il modello studente.

Questo processo consente al modello studente di acquisire una conoscenza meno sensibile ai piccoli cambiamenti negli input. Poiché la distillation porta ad avere confini decisionali più graduali, l'attaccante trova maggiori difficoltà nel calcolare perturbazioni minime che possano alterare il risultato della classificazione. In sostanza, rendendo il modello meno brusco nelle sue risposte, la defensive distillation contribuisce a mitigare l'efficacia degli attacchi basati sui gradienti, che spesso sfruttano le transizioni nette per individuare la direzione in cui manipolare gli input.

Difesa	**Caratteristiche**
Adversarial Training	Generazione di dati avversari e inclusione nel processo di addestramento del modello
Perturbation Control	Identificazione e correzione di input dannosi
Robustness by Certification	Garanzia formale di robustezza contro gli attacchi avversari
Distillation	Addestra un modello studente a imitare l'output probabilistico di un modello insegnante

Table 3: Sintesi delle differenze delle strategie difensive

La tabella 3 ci fornisce una sintesi delle principali caratteristiche delle strategie difensive descritte. La robustezza di un modello di intelligenza artificiale è determinata da molteplici fattori interconnessi, che vanno ben oltre la mera scelta delle strategie difensive. Elementi cruciali includono il *pre-processing* dei dati, la selezione dell'architettura, le metodologie di *training*, nonché le fasi di *fine-tuning* e *alignment*. Nonostante i significativi progressi nella ricerca sulla sicurezza dei modelli, le strategie difensive attualmente documentate in letteratura presentano ancora notevoli limitazioni in termini di efficacia globale. L'evidenza empirica dimostra che la maggior parte delle tecniche di difesa risulta efficace solo in scenari specifici e contro un sottoinsieme limitato di perturbazioni possibili. Questa vulnerabilità intrinseca è riconducibile principalmente a due fattori: da un lato, l'implementazione di sistemi di difesa più robusti richiederebbe risorse computazionali proibitive; dall'altro, persino in presenza di ingenti risorse computazionali, persisterebbero vettori di attacco in grado di compromettere l'integrità del modello. (OpenAI, 2017) nella ricerca *Attacking machine learning with adversarial examples*, consolida quanto detto in precedenza, citando:

> *"Adversarial examples are also hard to defend against because they require machine learning models to produce good outputs for every possible input. Most of the time, machine learning models work very well but only work on a very small amount of all the many possible inputs they might encounter."*
>
> *"Every strategy we have tested so far fails because it is not adaptive: it may block one kind of attack, but it leaves another vulnerability open to an attacker who knows about the defense being used. Designing a defense that can protect against a powerful, adaptive attacker is an important research area."*

4 Casi Applicativi in NLP

In questo capitolo, esamineremo due casi di studio che illustrano l'applicazione di tecniche di attacco su modelli con architetture fondamentalmente diverse: un classificatore tradizionale, e un modello neurale. Nello specifico, i casi presi in esame fanno parte di una più ampia raccolta, tratti dalla documentazione del framework TextAttack (Morris et al., 2020), del quale è già stata fornita una breve descrizione nel capitolo 1. Questa analisi comparativa ci permetterà di evidenziare se e in che modo le vulnerabilità e le strategie di attacco varino in base all'architettura sottostante. Al fine di garantire una comparazione rigorosa ed esaustiva, entrambi i casi di studio sono stati trattati nell'ambito della sentiment analysis, utilizzando dataset costituiti da recensioni cinematografiche. Vedremo i dataset più nello specifico nei sottocapitoli dedicati a ciascun caso applicativo.

4.1 Modello Non Neurale

Il primo caso vede l'utilizzo della libreria di machine learnig *scikit-learn*, per le fasi di training e di attacco del modello.

4.1.1 Dataset

La presente analisi si basa sul dataset *rotten_tomatoes*, reso disponibile attraverso la piattaforma HuggingFace. Il dataset comprende oltre 10.000 recensioni cinematografiche in lingua inglese, rappresentate in formato testuale. La variabile target, che identifica il sentiment, è codificata in forma binaria, dove i valori positivi (1) e negativi (0) sono equamente distribuiti, definendo il dataset come perfettamente bilanciato. La partizione dei dati è stata predefinita secondo una distribuzione standard: 80% dedicato al training set, 10% al validation set e il restante 10% al test set.

4.1.2 Pre-Processing

Dopo aver caricato e organizzato il dataset in un DataFrame, dove ogni recensione e la sua etichetta di sentiment vengono sistematizzate, viene sviluppata una pipeline per il pre-processamento e normalizzazione del testo.

Il testo di ciascuna recensione è sottoposto a una pulizia preliminare tramite REGEX che rimuove tutti i caratteri non alfabetici, garantendo così una base omogenea per le analisi. Successivamente, il testo pulito viene *tokenizzato* grazie a una funzione specifica che divide le recensioni in singole

unità lessicali. Questa operazione è affiancata da un processo di *stemming*, che riduce ogni parola alla sua radice, contribuendo a standardizzare e normalizzare il lessico.

Oltre a queste operazioni sul testo, vengono generate delle feature aggiuntive che arricchiscono ulteriormente il dataset. In particolare, viene calcolata la lunghezza di ogni recensione in termini di numero di token e la lunghezza media delle parole, entrambe normalizzate tramite una procedura di scaling. Questi indicatori strutturali forniscono al modello ulteriori elementi informativi che possono contribuire a migliorare la capacità discriminante durante la fase di classificazione.

Il passo successivo prevede la trasformazione del testo in una rappresentazione numerica, fondamentale per il successivo addestramento del modello. A tal fine, vengono impiegati due approcci distinti. Il primo approccio utilizza una tecnica di Bag-of-Words, che tramite un CountVectorizer trasforma il testo in un insieme di frequenze, tenendo in considerazione unigrammi, bigrammi e trigrammi, e limitando il vocabolario alle 100 feature più frequenti, escludendo al contempo le stop words in lingua inglese. Il secondo approccio si basa sul TfidfVectorizer, che pur mantenendo una configurazione analoga per quanto riguarda l'uso degli n-grammi e la dimensione massima del vocabolario, calcola per ogni termine un peso che tiene conto sia della frequenza locale sia della sua rilevanza nel corpus complessivo, permettendo così di evidenziare i termini distintivi.

4.1.3 TRAINING

Successivamente alle operazioni svolte nella fase di pre-processing, si procede all'addestramento di un modello di classificazione basato su *logistic regression*. Il classificatore viene configurato con un parametro di regolarizzazione $C = 30$ e un limite massimo di 200 iterazioni ($max\_iter = 200$). L'apprendimento supervisionato viene eseguito sulla partizione di training, mentre la validazione delle prestazioni avviene sulla partizione di test disponibile. La valutazione del modello viene condotta calcolando l'accuratezza su entrambi gli insiemi di dati e generando un report dettagliato che include le principali metriche di performance per task di classificazione.

4.1.4 ATTACCO

La strategia di attacco adottata è quella proposta dalla ricetta black-box TextFooler, tecnica affrontata nel capitolo precedente, che si basa sulla sostituzione di parole sfruttando lo spazio degli embedding. In pratica, per ogni parola del testo vengono individuati sinonimi semanticamente simili, limitando il numero di possibili sostituzioni con il parametro $max\_candidates = 50$, in modo da controllare l'espansione dello spazio di ricerca. La trasformazione applicata, infatti, sostituisce le parole con quelle individuate tramite il calcolo dei vicini più prossimi nell'embedding, assicurando al contempo che il significato originale del testo rimanga inalterato per la maggior parte, sebbene la classificazione del modello venga modificata. L'attacco è eseguito in modalità *untargeted*, ma come discusso in precedenza, data la natura binaria del task di classificazione, entrambe le tipologie avrebbero prodotto lo stesso risultato. Inoltre, sono stati impostati dei limiti per assicurare che la modifica non alteri drasticamente il senso originale del testo, cercando di mantenere la leggibilità e la coerenza; tra questi troviamo limiti sulla similarità coseno o vincoli su POS. L'intero attacco viene eseguito su dieci esempi estratti dal dataset, nei quali TextAttack genera versioni perturbate dei testi e monitora come questi cambiamenti influenzino la predizione, registrando metriche come la percentuale di parole modificate e il numero medio di query effettuate al modello.

4.1.5 RISULTATI

I risultati ottenuti evidenziano alcuni punti chiave sulle vulnerabilità critiche del modello. Il tasso di successo (ASR), metrica descritta nel capitolo 2.3, è del 100% per gli esempi attaccati (5 attacchi riusciti su 5 eseguiti), anche se va notato che sono stati saltati altri 5 esempi per cui l'attacco non è stato eseguito, probabilmente a causa dei vincoli o della difficoltà nell'applicare la modifica senza alterare eccessivamente il significato. L'accuratezza originale del modello era del 50% e scende allo 0% dopo l'attacco, evidenziando una vulnerabilità completa alle perturbazioni.

A questi preoccupanti valori si aggiunge il fatto che solo il 6.08% delle parole viene modificato in media, il che significa che piccole variazioni sono sufficienti per indurre errori di classificazione. Questa fragilità evidenzia come modelli basati su rappresentazioni semplici (come bag-of-words) siano particolarmente suscettibili a piccoli cambiamenti, che sfruttano la mancanza di robustezza semantica. Tuttavia, i risultati evidenziano anche la complessità del processo, con circa 67.6 query

per esempio, l'attacco richiede diverse valutazioni del modello, il che può essere un indicatore del costo computazionale del processo in scenari reali.

È necessario evidenziare che il modello è stato implementato senza l'integrazione di alcuna strategia o meccanismo di difesa contro attacchi avversari, elemento che contribuisce significativamente all'elevata efficacia dell'attacco condotto. L'assenza di contromisure difensive rappresenta infatti un fattore critico nella vulnerabilità del sistema alle perturbazioni malevole.

4.2 Modello Neurale

The text must be confined within a rectangle 5.5 inches (33 picas) wide and 9 inches (54 picas) long. The left margin is 1.5 inch (9 picas). Use 10 point type with a vertical spacing of 11 points. Times New Roman is the preferred typeface throughout. Paragraphs are separated by 1/2 line space, with no indentation.

Paper title is 17 point, in small caps and left-aligned. All pages should start at 1 inch (6 picas) from the top of the page.

Authors' names are set in boldface, and each name is placed above its corresponding address. The lead author's name is to be listed first, and the co-authors' names are set to follow. Authors sharing the same address can be on the same line.

Please pay special attention to the instructions in section 6 regarding figures, tables, acknowledgments, and references.

There will be a strict upper limit of 8 pages for the main text of the initial submission, with unlimited additional pages for citations. Note that the upper page limit differs from last year!Authors may use as many pages of appendices (after the bibliography) as they wish, but reviewers are not required to read these. During the rebuttal phase and for the camera ready version, authors are allowed one additional page for the main text, for a strict upper limit of 9 pages.

5 HEADINGS: FIRST LEVEL

First level headings are in small caps, flush left and in point size 12. One line space before the first level heading and 1/2 line space after the first level heading.

5.1 HEADINGS: SECOND LEVEL

Second level headings are in small caps, flush left and in point size 10. One line space before the second level heading and 1/2 line space after the second level heading.

5.1.1 HEADINGS: THIRD LEVEL

Third level headings are in small caps, flush left and in point size 10. One line space before the third level heading and 1/2 line space after the third level heading.

6 CITATIONS, FIGURES, TABLES, REFERENCES

These instructions apply to everyone, regardless of the formatter being used.

6.1 CITATIONS WITHIN THE TEXT

Citations within the text should be based on the `natbib` package and include the authors' last names and year (with the "et al." construct for more than two authors). When the authors or the publication are included in the sentence, the citation should not be in parenthesis using `\citet{}` (as in "See **?** for more information."). Otherwise, the citation should be in parenthesis using `\citep{}` (as in "Deep learning shows promise to make progress towards AI (**?**).").

The corresponding references are to be listed in alphabetical order of authors, in the REFERENCES section. As to the format of the references themselves, any style is acceptable as long as it is used consistently.

6.2 FOOTNOTES

Indicate footnotes with a number[3] in the text. Place the footnotes at the bottom of the page on which they appear. Precede the footnote with a horizontal rule of 2 inches (12 picas).[4]

[3] Sample of the first footnote

[4] Sample of the second footnote

Table 4: Sample table title

PART	DESCRIPTION
Dendrite	Input terminal
Axon	Output terminal
Soma	Cell body (contains cell nucleus)

6.3 FIGURES

All artwork must be neat, clean, and legible. Lines should be dark enough for purposes of reproduction; art work should not be hand-drawn. The figure number and caption always appear after the figure. Place one line space before the figure caption, and one line space after the figure. The figure caption is lower case (except for first word and proper nouns); figures are numbered consecutively.

Make sure the figure caption does not get separated from the figure. Leave sufficient space to avoid splitting the figure and figure caption.

You may use color figures. However, it is best for the figure captions and the paper body to make sense if the paper is printed either in black/white or in color.

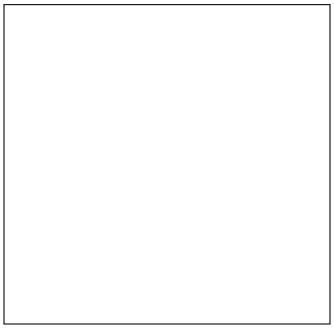

Figure 9: Sample figure caption.

6.4 TABLES

All tables must be centered, neat, clean and legible. Do not use hand-drawn tables. The table number and title always appear before the table. See Table 4.

Place one line space before the table title, one line space after the table title, and one line space after the table. The table title must be lower case (except for first word and proper nouns); tables are numbered consecutively.

7 DEFAULT NOTATION

In an attempt to encourage standardized notation, we have included the notation file from the textbook, *Deep Learning* **?** available at `https://github.com/goodfeli/dlbook_notation/`. Use of this style is not required and can be disabled by commenting out `math_commands.tex`.

Numbers and Arrays

a	A scalar (integer or real)
$\boldsymbol{a}$	A vector
$\boldsymbol{A}$	A matrix
$\mathsf{\mathbf{A}}$	A tensor
$\boldsymbol{I}_n$	Identity matrix with n rows and n columns
$\boldsymbol{I}$	Identity matrix with dimensionality implied by context
$\boldsymbol{e}^{(i)}$	Standard basis vector $[0, \dots, 0, 1, 0, \dots, 0]$ with a 1 at position i
$\text{diag}(\boldsymbol{a})$	A square, diagonal matrix with diagonal entries given by $\boldsymbol{a}$
a	A scalar random variable
$\mathbf{a}$	A vector-valued random variable
$\mathbf{A}$	A matrix-valued random variable

Sets and Graphs

$\mathbb{A}$	A set
$\mathbb{R}$	The set of real numbers
$\{0, 1\}$	The set containing 0 and 1
$\{0, 1, \dots, n\}$	The set of all integers between 0 and n
$[a, b]$	The real interval including a and b
$(a, b]$	The real interval excluding a but including b
$\mathbb{A} \backslash \mathbb{B}$	Set subtraction, i.e., the set containing the elements of $\mathbb{A}$ that are not in $\mathbb{B}$
$\mathcal{G}$	A graph
$Pa_{\mathcal{G}}(\mathrm{x}_i)$	The parents of x_i in $\mathcal{G}$

Indexing

a_i	Element i of vector $\boldsymbol{a}$, with indexing starting at 1
a_{-i}	All elements of vector $\boldsymbol{a}$ except for element i
$A_{i,j}$	Element i, j of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{i,:}$	Row i of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{:,i}$	Column i of matrix $\boldsymbol{A}$
$\mathsf{\mathbf{A}}_{i,j,k}$	Element (i, j, k) of a 3-D tensor $\mathsf{\mathbf{A}}$
$\mathsf{\mathbf{A}}_{:,:,i}$	2-D slice of a 3-D tensor
a_i	Element i of the random vector $\mathbf{a}$

Calculus

$\frac{dy}{dx}$	Derivative of y with respect to x
$\frac{\partial y}{\partial x}$	Partial derivative of y with respect to x
$\nabla_{\boldsymbol{x}} y$	Gradient of y with respect to $\boldsymbol{x}$
$\nabla_{\boldsymbol{X}} y$	Matrix derivatives of y with respect to $\boldsymbol{X}$
$\nabla_{\mathbf{X}} y$	Tensor containing derivatives of y with respect to $\mathbf{X}$
$\frac{\partial f}{\partial \boldsymbol{x}}$	Jacobian matrix $\boldsymbol{J} \in \mathbb{R}^{m \times n}$ of $f : \mathbb{R}^n \to \mathbb{R}^m$
$\nabla^2_{\boldsymbol{x}} f(\boldsymbol{x})$ or $\boldsymbol{H}(f)(\boldsymbol{x})$	The Hessian matrix of f at input point $\boldsymbol{x}$
$\int f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral over the entire domain of $\boldsymbol{x}$
$\int_{\mathbb{S}} f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral with respect to $\boldsymbol{x}$ over the set $\mathbb{S}$

Probability and Information Theory

$P(\mathrm{a})$	A probability distribution over a discrete variable
$p(\mathrm{a})$	A probability distribution over a continuous variable, or over a variable whose type has not been specified
$\mathrm{a} \sim P$	Random variable a has distribution P
$\mathbb{E}_{\mathrm{x} \sim P}[f(x)]$ or $\mathbb{E} f(x)$	Expectation of $f(x)$ with respect to $P(\mathrm{x})$
$\mathrm{Var}(f(x))$	Variance of $f(x)$ under $P(\mathrm{x})$
$\mathrm{Cov}(f(x), g(x))$	Covariance of $f(x)$ and $g(x)$ under $P(\mathrm{x})$
$H(\mathrm{x})$	Shannon entropy of the random variable x
$D_{\mathrm{KL}}(P \Vert Q)$	Kullback-Leibler divergence of P and Q
$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution over $\boldsymbol{x}$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$

Functions

$f : \mathbb{A} \to \mathbb{B}$	The function f with domain $\mathbb{A}$ and range $\mathbb{B}$
$f \circ g$	Composition of the functions f and g
$f(\boldsymbol{x}; \boldsymbol{\theta})$	A function of $\boldsymbol{x}$ parametrized by $\boldsymbol{\theta}$. (Sometimes we write $f(\boldsymbol{x})$ and omit the argument $\boldsymbol{\theta}$ to lighten notation)
$\log x$	Natural logarithm of x
$\sigma(x)$	Logistic sigmoid, $\frac{1}{1 + \exp(-x)}$
$\zeta(x)$	Softplus, $\log(1 + \exp(x))$
$\Vert \boldsymbol{x} \Vert_p$	L^p norm of $\boldsymbol{x}$
$\Vert \boldsymbol{x} \Vert$	L^2 norm of $\boldsymbol{x}$
x^+	Positive part of x, i.e., $\max(0, x)$
$\mathbf{1}_{\mathrm{condition}}$	is 1 if the condition is true, 0 otherwise

8 FINAL INSTRUCTIONS

Do not change any aspects of the formatting parameters in the style files. In particular, do not modify the width or length of the rectangle the text should fit into, and do not change font sizes (except perhaps in the REFERENCES section; see below). Please note that pages should be numbered.

9 PREPARING POSTSCRIPT OR PDF FILES

Please prepare PostScript or PDF files with paper size "US Letter", and not, for example, "A4". The -t letter option on dvips will produce US Letter files.

Consider directly generating PDF files using `pdflatex` (especially if you are a MiKTeX user). PDF figures must be substituted for EPS figures, however.

Otherwise, please generate your PostScript and PDF files with the following commands:

```
dvips mypaper.dvi -t letter -Ppdf -G0 -o mypaper.ps
ps2pdf mypaper.ps mypaper.pdf
```

9.1 MARGINS IN LATEX

Most of the margin problems come from figures positioned by hand using `\special` or other commands. We suggest using the command `\includegraphics` from the graphicx package. Always specify the figure width as a multiple of the line width as in the example below using .eps graphics

```
\usepackage[dvips]{graphicx} ...
\includegraphics[width=0.8\linewidth]{myfile.eps}
```

or

```
\usepackage[pdftex]{graphicx} ...
\includegraphics[width=0.8\linewidth]{myfile.pdf}
```

for .pdf graphics. See section 4.4 in the graphics bundle documentation (`http://www.ctan.org/tex-archive/macros/latex/required/graphics/grfguide.ps`)

A number of width problems arise when LaTeX cannot properly hyphenate a line. Please give LaTeX hyphenation hints using the `\-` command.

AUTHOR CONTRIBUTIONS

If you'd like to, you may include a section for author contributions as is done in many journals. This is optional and at the discretion of the authors.

ACKNOWLEDGMENTS

Use unnumbered third level headings for the acknowledgments. All acknowledgments, including those to funding agencies, go at the end of the paper.

REFERENCES

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015. URL https://arxiv.org/abs/1412.6572.

Shreya Goyal, Sumanth Doddapaneni, Mitesh M. Khapra, and Balaraman Ravindran. A survey of adversarial defences and robustness in nlp, 2023. URL https://arxiv.org/abs/2203.06414.

Nodens Koren, Qiuhong Ke, Yisen Wang, James Bailey, and Xingjun Ma. Adversarial interaction attack: Fooling ai to misinterpret human intentions, 2021. URL https://arxiv.org/abs/2101.06704.

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp, 2020.

OpenAI. attacking-machine-learning-with-adversarial-examples, 2017. URL https://openai.com/index/attacking-machine-learning-with-adversarial-examples.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning, 2017. URL https://arxiv.org/abs/1602.02697.

Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. Adversarial attacks on deep learning models in natural language processing: A survey, 2019. URL https://arxiv.org/abs/1901.06796.

A Appendix

You may include other additional sections here.