UNIVERSITÀ DEGLI STUDI DI URBINO CARLO BO

Dipartimento di Scienze Pure e Applicate
Corso di Laurea Magistrale in Informatica Applicata

Tesi di Laurea

APPLICAZIONE DI TECNICHE DI AI NEURO-SIMBOLICA IN AMBITO SANITARIO: IDEAZIONE E VALUTAZIONE DI METODI PER L'INIEZIONE DI CONOSCENZA

Relatore:
Chiar.ma Prof.ssa Sara Montagna

Candidato:
Nunzio D'Amore

Correlatori:
Dott.ssa Christel Sirocchi

Anno Accademico 2024-2025

Ai miei genitori

Indice

Elenco delle figure

Elenco delle tabelle

Elenco Acronimi

AI Artificial Intelligence

IML Machine Learning Informato

KBANN Knowledge-Based Neural Networks

KINS Knowledge Injection via Network Structuring

LTN Logic Tensor Networks

ML Machine Learning

SKE Symbolic Knowledge Extraction

SKI Symbolic Knowledge Injection

STI Siamese Training Injection

TEFI train-extract-fix-inject

HN Hayashi e Nakano

DAG Duch, Adamczak e Grabczewski

Capitolo 1

Introduzione

Negli ultimi anni, l'Artificial Intelligence (AI) si è affermata come una delle tecnologie più rilevanti, rivoluzionando il modo di approciarsi a problemi complessi. I progressi di queste tecnologie con modelli di Machine Learning (ML) semplici e reti neurali profonde con metodi di apprendimento supervisionato e non supervisionato hanno permesso di raggiungere risultati straordinari in numerosi settori. In particolare, l'ambito medico-sanitario ha visto un incremento significativo nell'adozione di soluzioni basate sull'AI.

In questo ambito si riscontrano, però, diversi ostacoli che limitano l'applicazione di queste tecnologie. I dati sono spesso **limitati** poiché dispendiosi da recuperare in termini di tempo e risorse. Spesso i dati sono **sbilanciati**, sottorappresentando condizioni patologiche. Ulteriori difficoltà sorgono dal rischio che i dati possano **riflettere dinamiche sociali esistenti** penalizzando ingiustamente soggetti appartenenti a minoranze, compromettendo l'equità dei trattamenti. Un'altra criticità è il cosiddetto **"black-box problem"**: i modelli di AI più avanzati spesso forniscono risultati accurati, ma senza offrire una chiara interpretazione delle loro decisioni. Inoltre, l'adozione di questi strumenti dovrebbe comportare un reale miglioramento rispetto alle pratiche esistenti, restando comunque allineata alle linee guida consolidate. Questo è fondamentale sia per ragioni etiche che legali, al fine di evitare decisioni errate in situazioni in cui i metodi tradizionali forniscono già risposte adeguate. Tuttavia, i modelli basati su IA, pur ottenendo ottime prestazioni, possono individuare correlazioni nei dati che non riflettono necessariamente conoscenze già validate. Questo può disincentivare l'adozione di tali strumenti.

Le tradizionali tecniche di ML, pur essendo estremamente efficaci nel trovare legami complessi in grandi quantità di dati, presentano, quindi, alcune limitazioni intrinseche. Questi limiti sono particolarmente rilevanti in ambito sanitario, dove la comprensibilità e la trasparenza delle decisioni sono essenziali per garantire la fiducia degli operatori sanitari e la sicurezza dei pazienti.

Una possibile soluzione a queste problematiche, potrebbe risiedere nell'integrazione tra conoscenze simboliche e metodi di apprendimento automatico. I metodi di Symbolic Knowledge Injection (SKI) consentono di integrare nei modelli di ML conoscenze provenienti da esperti del dominio, linee guida mediche e altre fonti, opportunamente tradotte in forma simbolica. Questo approccio mira a migliorare i tempi di convergenza, la capacità di generalizzazione del modello e la sua aderenza alle linee guida, anche in situazioni complesse.

D'altro canto, i metodi di Symbolic Knowledge Extraction (SKE) permettono di estrapolare conoscenza simbolica dai modelli addestrati, con l'obiettivo di riutilizzarla nell'addestramento futuro o eventualmente di ricavare nuova conoscenza medica.

Questa tesi, si pone l'obiettivo di esplorare i principali metodi di SKI, analizzando il loro impatto sulle applicazioni di modelli di ML in ambito sanitario. Verranno testati metodi già conosciuti e consolidati, come Knowledge-Based Neural Networks (KBANN), Knowledge Injection via Network Structuring (KINS) e Logic Tensor Networks (LTN), evidenziando i benefici e carenze di ciascun approccio. Oltre a questi metodi, verranno esplorate nuove metodologie sviluppate nell'ambito di questo lavoro di ricerca.

Inoltre, si testerà la capacità dei metodi di SKI di migliorare la robustezza dei modelli a seguito di perturbazioni dei dati. Distingueremo tra perturbazioni già applicate a metodi di SKI in letteratura: feature noise, data dropping e label flipping; e una nuova perturbazione proposta in questo studio, la feature corruption.

Vedremo che i risultati dei metodi esistenti sono stati discreti, non apportando particolari miglioramenti al modello non educato dal punto di vista delle performance. KBANN si è rilevato comunque un valido strumento, migliorando la robustezza e la resistenza a metodi di perturbazione se applicato a dati semplici, ma con scarse capacità di generalizzazione su dati più complessi. Infine, tra i metodi testati si sono distinti LTN per i metodi di perturbazione esistenti e KINS per la feature corruption, migliorando la robustezza del modello non educato.

Presenteremo, inoltre, un nuovo metodo di SKI che basa il suo funzionamento sull'addestramento siamese, che consiste nella possibilità di addestrare lo stesso modello su più target contemporaneamente. Sfrutteremo questo metodo di addestramento per permettere al modello di generalizzare, e allo stesso tempo di rispettare la conoscenza.

Verranno eseguiti diversi test, con diverse configurazioni. Mostreremo, per

ognuna, pregi e difetti apportando miglioramenti graduali. Il metodo verrà testato prima utilizzando, come dati che rappresentano la conoscenza, quelli già presenti nel dataset di partenza, per poi sfruttare la data augmentation generando dati sulla base della conoscenza stessa. Questo ci permetterà di rappresentare in modo affidabile la conoscenza, apportando miglioramenti alla robustezza del modello, anche in casi in cui la conoscenza non viene descritta appropriatamente nei dati. Inoltre, modificheremo la pipeline di addestramento permettendo al modello di apprendere, attraverso una fase di preaddestramento, le relazioni che caratterizzano la conoscenza, per poi effettuare l'addestramento siamese, in cui si cerca di permettere al modello di generalizzare al meglio, tenendo alta la coerenza con la conoscenza. Infine, apporteremo miglioramenti modificando il range di generazione degli elementi e successivamente testeremo il modello eliminando quegli elementi del dataset che sono contrari alla conoscenza.

Vedremo che il nuovo metodo è riuscito ad avere performance migliori dei metodi esistenti e che riesce ad apportare miglioramenti specialmente dal punto di vista della robustezza, diminuendo, inoltre, il numero di epoche necessarie per convergere.

Il metodo è ancora inesplorato e bisognerà effettuare ancora numerose prove ed esperimenti per confermarne la validità, ma riesce comunque a dimostrare che la conoscenza può concretamente guidare i metodi di ML migliorandone le caratteristiche.

Capitolo 2

Contesto

2.1 Intelligenza artificiale, benefici e limiti

L'Artificial Intelligence (AI) è un campo dell'informatica che si propone di sviluppare sistemi in grado di replicare capacità tipiche della mente umana, come l'apprendimento, il ragionamento e il processo decisionale. Per raggiungere questo obiettivo, l'AI si avvale di numerosi metodi, tra cui modelli statistici come le reti neurali, che sono in grado di apprendere complesse relazioni a partire dai dati. Questo tipo di approccio rientra nell'ambito del Machine Learning (ML). L'apprendimento nell'AI consiste nell'ottimizzazione dei parametri di un modello statistico, basata sull'analisi di grandi quantità di dati che descrivono la realtà di interesse. Un'alternativa a questo approccio è il reinforcement learning, in cui il modello apprende interagendo con l'ambiente, ricevendo ricompense per le azioni corrette e penalizzazioni per quelle errate.

L'importanza crescente dell'AI è evidente e il suo impatto si estende ormai a tutti i settori, incluso quello medico-sanitario. Le sue applicazioni spaziano dalla diagnosi assistita al rilevamento precoce, dal supporto alle decisioni cliniche alla ricerca e allo sviluppo di nuovi farmaci. Inoltre, l'AI riveste un ruolo fondamentale nella medicina personalizzata, contribuendo alla definizione di terapie su misura e alla simulazione della risposta del paziente.

2.1.1 Benefici del Machine Learning

Il ML è fondamentale in queste applicazioni perché è in grado di analizzare e prendere decisioni in base a grandi quantità di dati, comprendendone le relazioni più complesse e nascoste, migliorando l'assistenza sanitaria, rendendola più efficiente, accessibile e personalizzata. Inoltre, questi sistemi sono dotati di grande flessibilità, che rende possibile adottare modelli di ML su un'ampia varietà di dati. Possiamo analizzare immagini per l'analisi di radiografie, risonanze magnetiche, TAC; testi quali referti medici e pubblicazioni scientifiche,

per estrarre informazioni rilevanti e supportare la diagnosi; segnali biometrici e fisiologici, come elettrocardiogrammi, elettroencefalogrammi e parametri vitali, per rilevare anomalie in tempo reale. Questa versatilità consente di integrare il ML in molteplici ambiti della medicina, migliorando la precisione, la rapidità e l'efficacia delle decisioni cliniche, per personalizzare trattamenti e prevedere predisposizioni a malattie.

2.1.2 Limiti del Machine Learning

I dati in ambito sanitario presentano particolari caratteristiche che complicano l'analisi e l'utilizzo di queste tecnologie. Diversi fattori contribuiscono a rendere la raccolta di dati in medicina particolarmente complessa e costosa: **l'evoluzione continua delle malattie** per cui le patologie sono in costante cambiamento, richiedendo un aggiornamento continuo dei dati per mantenere la rilevanza dei modelli predittivi; **il ritardo nella digitalizzazione delle istituzioni mediche** per cui la mancata adozione diffusa di sistemi digitali avanzati ostacola la raccolta e la condivisione efficiente dei dati; **i requisiti legali per la protezione dei dati sensibili** dato che le normative sulla privacy impongono restrizioni rigorose sulla condivisione e l'utilizzo dei dati medici, limitando le possibilità di sfruttare le fonti di dati esistenti.

I dataset sanitari sono dunque spesso soggetti a scarsità dei dati, dovuta alla dispendiosità in termini di tempo e risorse della raccolta di informazioni cliniche o a normative della privacy stringenti che limitano l'uso di dati per l'addestramento di modelli. Altri problemi derivano dalla mancanza di standardizzazione della raccolta delle informazioni che rende molto complicato integrare dati provenienti da fonti diverse, il che risulta in rumore o mancanza di dati. Altre fonti di rumore sono l'errore nelle misurazioni e l'errore umano. Di conseguenza, i dataset sanitari sono spesso limitati nella dimensione e soggetti a sbilanciamenti, per cui le condizioni patologiche sono spesso fortemente sottorappresentate rispetto ai casi normali. Di conseguenza, i dataset sanitari sono spesso limitati nella dimensione. Sono inoltre soggetti a sbilanciamenti poichè le con- dizioni patologiche sono spesso fortemente sottorappresentate rispetto ai casi normali.

Un ulteriore ostacolo è rappresentato dalla necessità di interpretabilità e trasparenza dei modelli. In ambito sanitario, le decisioni devono essere spiegabili e verificabili per aumentare la fiducia degli operatori sanitari e la sicurezza dei pazienti. Modelli di ML puramente black-box, ovvero modelli complessi e non interpretabili il cui processo decisionale interno è opaco agli esseri umani, non soddisfano queste esigenze, soprattutto in scenari critici come la diagnosi di malattie gravi o la pianificazione di terapie personalizzate. Quindi risulta particolarmente utile riuscire a comprendere come questi modelli effettuano le loro decisioni.

Un'altra problematica dei dati in ambito sanitario è il cosiddetto **bias**, per cui questi potrebbero riflettere delle dinamiche sociali che portano erroneamente a trattare e considerare in modo differente diverse popolazioni e minoranze, causando discriminazione.

Inoltre, è importante considerare la **coerenza con la conoscenza di dominio**, dato che non è per nulla scontato che i modelli di ML siano in grado di rispettare vincoli imposti da leggi naturali, normative o linee guida cliniche consolidate, con il rischio di generare decisioni non affidabili. Nelle prossime sezioni, approfondiremo questi limiti più nel dettaglio.

Numerosità e qualità dei dati

L'efficacia dei modelli di apprendimento automatico è intrinsecamente legata alla qualità e alla quantità dei dati utilizzati per il loro addestramento.

La quantità di dati si manifesta come un fattore determinante nella capacità di un modello di apprendere relazioni complesse. Modelli avanzati, in particolare quelli basati su architetture di deep learning, richiedono un ampio spettro di esempi per ottimizzare un vasto insieme di parametri. L'insufficienza di dati di addestramento conduce inevitabilmente al fenomeno dell'**overfitting**, in cui il modello si specializza eccessivamente nei dettagli specifici del dataset di addestramento, a discapito della generalizzazione, ovvero la sua abilità di effettuare previsioni su dati non precedentemente osservati.

Allo stesso modo, la qualità dei dati risulta fondamentale per le prestazioni del modello. La presenza di rumore, errori o distorsioni nei dati possono alterare le distribuzioni statistiche e compromettere la capacità del modello di identificare correttamente le relazioni che intercorrono tra i dati. Questa vulnerabilità è particolarmente evidente negli esperimenti di perturbazione dei dati, dove anche piccole variazioni nei dati possono generare fluttuazioni drastiche nelle previsioni del modello.

La sensibilità dei sistemi di apprendimento automatico a queste problematiche è intrinseca nella natura statistica di questi modelli, che si basano sull'identificazione di correlazioni e pattern nei dati per effettuare inferenze. Dati di scarsa qualità o quantità, producono rappresentazioni distorte della realtà e ne conseguono modelli inaffidabili.

Pertanto, la preparazione dei dati assume un ruolo critico nel processo di sviluppo di modelli di apprendimento automatico. Con il fine di ridurre l'effetto di queste problematiche, vengono messe in atto tecniche di pulizia e data augmentation, ovvero la generazione di dati sintetici.

Bias and equity

Un altro aspetto cruciale, strettamente legato alle caratteristiche dei dati sanitari, è quello del **bias ed equità** dei modelli di AI. Dati limitati o non rappresentativi possono introdurre bias, compromettendo l'equità dei modelli quando applicati a diverse popolazioni. Per bias si intende una tendenza sistematica a favorire o sfavorire determinate persone, gruppi o idee. Nel nostro caso, il bias si manifesta quando un modello produce risultati sistematicamente distorti a causa di dati o algoritmi imperfetti. Nel contesto clinico, ciò può tradursi in decisioni algoritmiche distorte che incidono negativamente sulla salute delle persone.

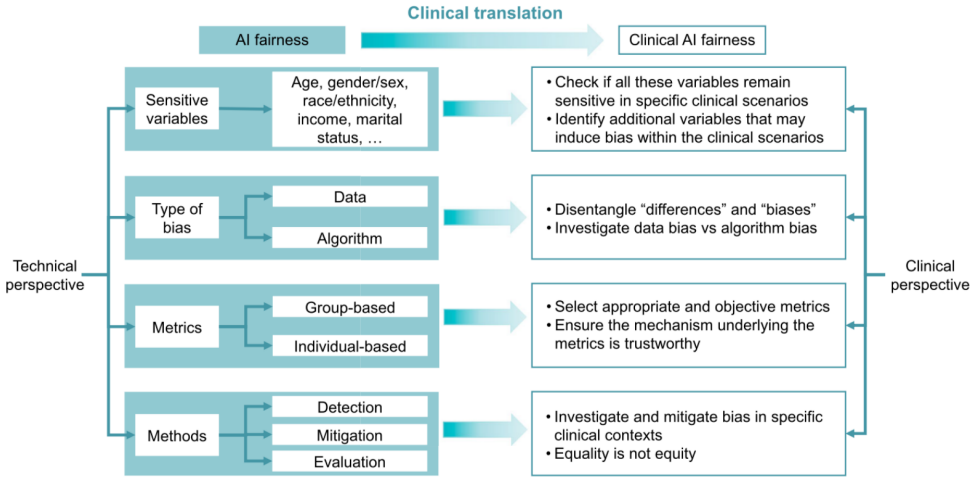


Figura 2.1: Modello concettuale verso l'equità clinica dell'AI. [1]

La figura 2.1 mostra un modello concettuale che affronta l'equità nell'AI da due prospettive: tecnica e clinica.

La **prospettiva tecnica** comprende l'attenzione alle **variabili sensibili** (come età, genere, razza, reddito e stato civile), che devono essere monitorate per evitare che i modelli perpetuino discriminazioni. I **bias nei dati** si manifestano quando questi non rappresentano correttamente la popolazione reale, per esempio a causa di sottorappresentazione di gruppi demografici, distorsioni dovute a pregiudizi sociali o informazioni mancanti. I **bias negli algoritmi**, invece, derivano da scelte progettuali, metriche di ottimizzazione scorrette o interazioni complesse tra variabili che producono risultati iniqui.

Per misurare e valutare l'equità dei modelli, si usano apposite **metriche**. Quelle basate sul gruppo confrontano le performance del modello tra diversi gruppi demografici, mentre quelle basate sull'individuo valutano la coerenza dei risultati tra individui simili. Esistono anche **metodi** specifici per rilevare i bias,

mitigarli e valutarne l'andamento nel tempo.

La **prospettiva clinica** applica questi principi al settore sanitario. Considera, ad esempio, la verifica dell'effettiva sensibilità delle variabili in contesti clinici specifici, valutando se il loro utilizzo sia appropriato e privo di pregiudizi. È inoltre importante identificare variabili aggiuntive che potrebbero introdurre bias, distinguere tra differenze biologiche o sociali e reali distorsioni, selezionare metriche affidabili per monitorare l'equità e riconoscere che equità e uguaglianza non coincidono necessariamente. L'equità richiede infatti di adattare le cure alle specificità individuali, non semplicemente trattare tutti allo stesso modo.

Nel settore sanitario, i bias possono avere conseguenze particolarmente gravi, portando a diagnosi errate, trattamenti iniqui e disparità nell'accesso alle cure. Pertanto, è fondamentale adottare un approccio che permetta di evitare queste disuguaglianze.

Trasparenza e spiegabilità

Un'ulteriore sfida nell'applicazione di queste tecnologie nel settore sanitario è rappresentata dalla necessità di conformarsi a normative come il General Data Protection Regulation (GDPR) e l'European AI Act. Questi regolamenti, nati per tutelare i diritti dei cittadini, richiedono che le decisioni automatizzate che li riguardano siano accompagnate da spiegazioni chiare e accessibili, promuovendo trasparenza e responsabilità nei sistemi basati su IA.

Si delineano, quindi, due approcci principali: l'explainability by design e l'explainability post-hoc.

L'**explainability by design** mira a rendere i sistemi intelligenti interpretabili sin dalla fase di progettazione, adottando modelli intrinsecamente comprensibili - come gli alberi decisionali - oppure imponendo vincoli intelligibili al comportamento dei modelli predittivi, anche quando questi includono componenti black-box [19].

L'**explainability post-hoc**, invece, affronta il problema dell'interpretabilità ex-post, applicando una gamma di strategie per spiegare il funzionamento di modelli opachi. Tra queste vi sono: la generazione di spiegazioni testuali o visive, la suddivisione dello spazio delle soluzioni in sottospazi più semplici per abilitare spiegazioni locali, l'identificazione di esempi rappresentativi per illustrare le relazioni apprese, e l'assegnazione di punteggi di rilevanza alle caratteristiche per evidenziarne l'importanza [20]. Un'ulteriore strategia consiste nella semplificazione del modello, che prevede la costruzione di un sistema

alternativo più semplice ma funzionalmente simile all'originale, riducendone la complessità. Questa tecnica spesso si avvale di metodi di estrazione di conoscenza simbolica [22].

Coerenza con conoscenza di dominio

Un approccio puramente data-driven può rivelarsi inadeguato nel rispettare vincoli imposti da leggi naturali, normative o linee guida di sicurezza, elementi fondamentali per garantire degli strumenti affidabili.

Spesso, poi, i modelli di ML mostrano una limitata capacità nel riconoscere le relazioni intrinseche tra le variabili, portando a decisioni influenzate da fattori confondenti o latenti privi di una valida interpretazione fisica.

I fattori confondenti sono variabili non direttamente incluse nei dati che influenzano sia l'input che l'output. Crea quindi false relazioni causali in realtà dovure al fattore confondente. Un esempio può essere in uno studio che cerca di determinare se il consumo di caffè è correlato al rischio di malattie cardiache. Il fumo potrebbe essere un fattore confondente, poiché le persone che bevono caffè tendono anche a fumare. Il fumo, reale fattore di rischio per le malattie cardiache, se non considerato nello studio potrebbe portare erroneamente a considerare il caffè come sostanza pericolosa.

I fattori latenti non sono direttamente variabili osservabili o misurabili, ma le influenzano, e sono spesso concetti astratti o fenomeni complessi. Ad esempio possono essere fattori psicologici o di soddisfazione generale che influenzano il comportamento di un cliente, anche se non direttamente misurabile.

Ne segue che l'aderenza a linee guida cliniche consolidate, frutto di secoli di ricerca, non è sempre garantita, nonostante l'elevata accuratezza che tali modelli possano raggiungere. Questo, specialmente in contesto sanitaro, comporta significativi rischi nell'adozione pratica di queste tecnologie, in quanto modelli che non si allineano con protocolli noti possono generare preoccupazioni legate a errori potenzialmente evitabili, comportando conseguenti rischi etici e legali.

2.2 Machine Learning Informato

Una possibile soluzione a queste problematiche è rappresentata da quello che viene chiamato **Machine Learning Informato (IML)** [5], ovvero l'integrazione tra ML e conoscenza simbolica. Per **conoscenza simbolica** si intende un insieme di informazioni strutturate e formalizzate che descrivono la realtà di riferimento. Queste informazioni possono essere rappresentate utilizzando **diversi formalismi**, come la logica proposizionale, la logica del primo ordine,

relazioni probabilistiche, equazioni algebriche, equazioni differenziali, feedback umano, e tanto altro. Più queste informazioni sono rappresentate in modo formale, più è semplice integrarle nei modelli di ML. Qualora questa conoscenza fosse preesistente e non dipendente dal processo di addestramento, possiamo chiamarla conoscenza pregressa.

Questa metodologia è un approccio che mira a combinare il ragionamento simbolico, tipico dei sistemi esperti, ovvero quei sistemi che utilizzano regole logiche e sistemi di inferenza per prendere decisioni, con le capacità di apprendimento e generalizzazione dei modelli statistici. Questo paradigma nasce dalla consapevolezza che i sistemi puramente data-driven, come anticipato, hanno difficoltà nel generalizzare in presenza di dati scarsi o rumorosi.

L'integrazione di conoscenza pregressa nei modelli di apprendimento automatico offre numerosi vantaggi. Innanzitutto, **migliora la generalizzazione**, rendendo i modelli più robusti anche in presenza di dati sparsi o rumorosi. Inoltre, **riduce la dipendenza dai dati**, sfruttando la conoscenza del dominio per limitare la necessità di ampi dataset etichettati, spesso costosi in termini di tempo e risorse.

Un altro aspetto cruciale è l'**incremento dell'interpretabilità**: la maggiore trasparenza del modello semplifica la comprensione delle decisioni, rafforzando la fiducia nell'affidabilità dei risultati. L'**ottimizzazione dell'efficienza** rappresenta un ulteriore beneficio, accelerando la convergenza e riducendo il consumo di risorse computazionali.

Infine, l'integrazione di conoscenza consente di **affrontare meglio le difficoltà dei modelli puramente data-driven** nella gestione di valori anomali, eventi rari o casi limite. In questi contesti, la conoscenza pregressa si rivela essenziale per fornire un quadro interpretativo significativo e migliorare la capacità del modello di affrontare situazioni non frequenti ma rilevanti.

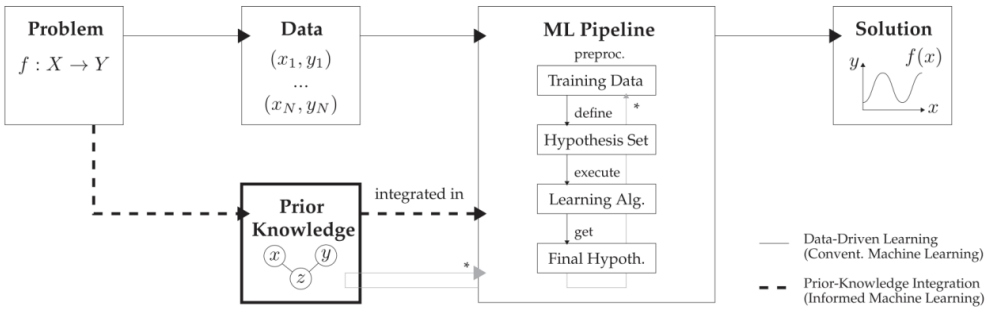


Figura 2.2: Flusso del Machine Learning Informato [4]

La figura 2.2 mostra il flusso delle informazioni nella pipeline del IML. Le informazioni che caratterizzano un problema provengono da due fonti differenti, quali i dati e la conoscenza pregressa.

Nel IML sono stati sviluppati diversi metodi in base a tre caratteristiche principali: la **sorgente** dell'informazione che viene integrata, la **rappresentazione** della conoscenza e **dove viene integrata** nella pipeline di addestramento.

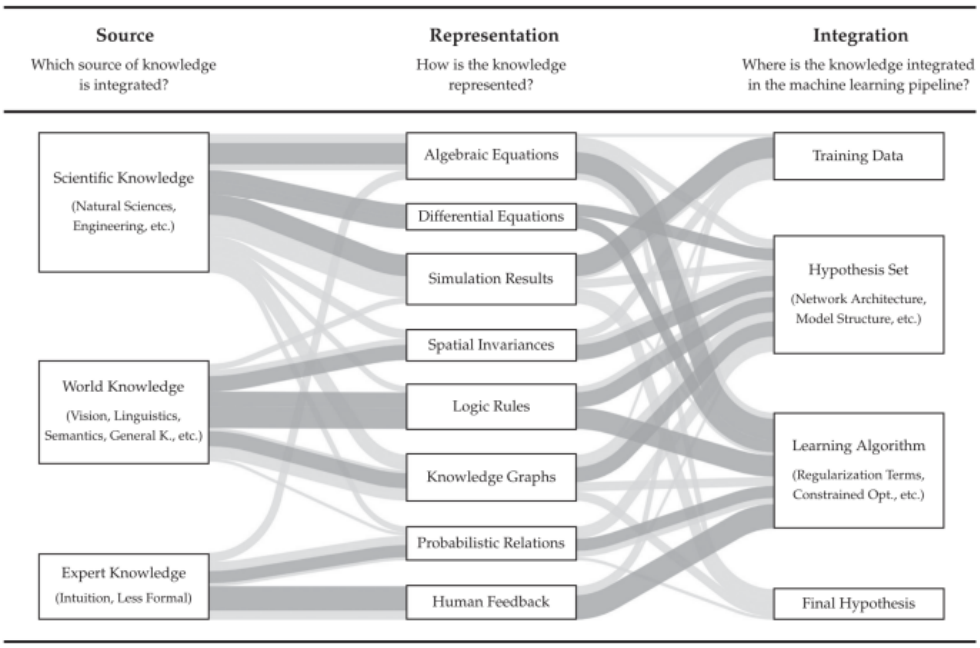


Figura 2.3: Caratteristiche di classificazione degli approcci di IML [4]

A partire da questo punto, concentreremo la nostra attenzione sulla conoscenza degli esperti come fonte di informazione, sulle regole logiche come metodo di rappresentazione e sugli algoritmi di apprendimento come punto di integrazione all'interno del processo di machine learning. Esploreremo, quindi, come le intuizioni e le competenze umane possano essere tradotte in regole formali e utilizzate per migliorare l'efficacia degli algoritmi di apprendimento.

2.2.1 Strategie di integrazione della conoscenza pregressa

L'integrazione della conoscenza pregressa può avvenire attraverso diverse modalità, ciascuna con specifiche caratteristiche e applicazioni. Una possibilità è l'incorporazione di **regole logiche ed equazioni algebriche** nei modelli, utilizzandole come vincoli per influenzare la funzione di perdita e guidare l'apprendimento verso soluzioni più coerenti con la conoscenza pregressa. Un'altra

strategia consiste nell'impiego dei **grafi di conoscenza**, che forniscono informazioni sulle relazioni tra entità e arricchiscono i modelli neurali, permettendo loro di cogliere connessioni complesse e migliorare la generalizzazione. Infine, le **simulazioni fisiche** possono essere sfruttate per generare dati di addestramento aggiuntivi, ampliando la copertura dei dati e potenziando la capacità dei modelli di adattarsi a situazioni non direttamente osservate.

Questa diversità di approcci ha portato anche a diverse terminologie, come *Physics-Informed Deep Learning* [6], *Physics-Guided Neural Networks* [7] e *Neurosymbolic AI*.

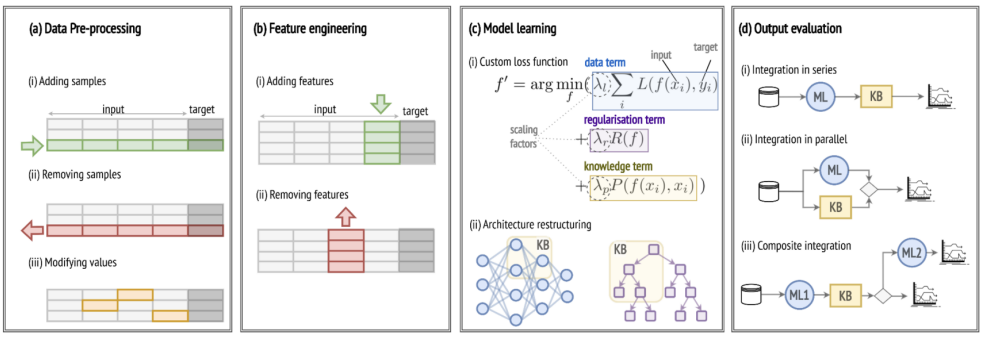


Figura 2.4: Strategie di integrazione della conoscenza pregressa [8]

La figura 2.4 mostra le principali strategie adottate per integrare la conoscenza simbolica nei modelli di ML. Questa integrazione può avvenire in diverse fasi. Il **preprocessing dei dati** rappresenta una prima possibilità, includendo operazioni come la discretizzazione, l'imputazione di dati mancanti o la generazione di dati sintetici basata sulla conoscenza diretta.

Un'altra strategia è il **feature engineering**, che consiste nell'aggiungere nuove feature basate su modelli matematici o inferenza logica dalla base di conoscenza. Questo approccio può anche mirare alla riduzione della dimensionalità, comprimendo più feature in pochi indici e migliorando contemporaneamente accuratezza e interpretabilità. Inoltre, attraverso la feature selection, è possibile selezionare solo le feature più rilevanti, eliminando quelle che potrebbero compromettere le performance del modello.

La **modifica del processo di addestramento** rappresenta un ulteriore metodo di integrazione. Ciò può avvenire alterando la funzione di perdita affinché il modello rispetti i vincoli della conoscenza o intervenendo sulla struttura della rete e sui suoi iperparametri.

Infine, la **valutazione dell'output** consente di confrontare o validare i risultati del modello con la conoscenza esistente. L'output può essere combinato in

serie o in parallelo con quello derivante dalla conoscenza simbolica, oppure può essere integrato in un framework composito che utilizza metodi di aggregazione per sfruttare sinergicamente più modelli.

2.2.2 Applicazioni del Machine Learning Informato

L'IML si è dimostrato efficace nell'analisi delle immagini e dei segnali medici, sfruttando la conoscenza del dominio per migliorare i risultati, in diverse occasioni.

Analisi delle Immagini

L'integrazione della conoscenza esperta nell'analisi delle immagini mediche ha portato a significativi progressi. Tecniche di segmentazione guidate da regole definite da esperti sono ampiamente utilizzate, ad esempio nell'analisi delle immagini del fondo oculare e delle risonanze magnetiche [9]. Inoltre, le reti neurali convoluzionali (CNN) sono spesso combinate con feature progettate manualmente per migliorare il rilevamento delle lesioni, come nel caso delle lesioni cervicali e delle immagini del fondo oculare [10] [11]. L'uso della regolarizzazione aiuta a garantire una maggiore coerenza con la conoscenza anatomica, migliorando l'affidabilità delle analisi su tomografie computerizzate (CT) e immagini PET [12] [13].

Analisi dei Segnali

Nell'elaborazione dei segnali medici, la conoscenza esperta gioca un ruolo fondamentale nel miglioramento della qualità dei dati. Tecniche di denoising guidate da esperti sono utilizzate per migliorare la qualità dei segnali ECG e PCG, riducendo il rumore e aumentando la precisione diagnostica [14] [15].

Impatto Generale

L'impiego dell'Apprendimento Automatico nell'analisi delle immagini e dei segnali medici consente di aumentare la precisione e la rilevanza clinica dei modelli. L'integrazione della conoscenza esperta in diverse fasi del processo, dalla raccolta dei dati alla loro pre-elaborazione fino alla generazione dei risultati del modello, garantisce un approccio più robusto ed efficace nell'ambito medico.

2.3 Intelligenza Artificiale Neurosimbolica

Tra le diverse branche dell'AI possiamo, quindi, distinguerne due.

L'**AI simbolica** si basa sul ragionamento attraverso rappresentazioni ad alto livello e leggibili dall'uomo, come la logica, le regole e i grafi di conoscenza.

Questo approccio utilizza rappresentazioni simboliche, ossia qualsiasi linguaggio comprensibile e interpretabile sia dagli esseri umani che dai computer, per codificare esplicitamente la conoscenza e applicare inferenza logica per prendere decisioni e svolgere ragionamenti.

Il **ML** è la branca dell'AI che si basa su modelli statistici in grado di apprendere dai dati relazioni complesse, sulla base delle quali prendere decisioni. Le informazioni derivate dai modelli di ML sono tipicamente rappresentate in forma sub-simbolica (ad esempio, numerica). Queste rappresentazioni, sebbene potenti, vengono manipolate attraverso operazioni algebriche o differenziali e sono intrinsecamente difficili da interpretare e comprendere per gli esseri umani. Poiché l'interpretabilità dei modelli predittivi e la comprensibilità dei loro risultati diventano sempre più critiche nelle applicazioni basate sui dati, i metodi di apprendimento automatico soffrono di limitazioni significative a causa della loro natura opaca e sub-simbolica.

Dall'integrazione di queste due branche deriva l'**AI Neurosimbolica**, che combina i punti di forza di entrambi gli approcci (figura 2.5). Questa integrazione unisce la capacità di interpretazione e ragionamento dei metodi simbolici con l'apprendimento basato sui dati dei modelli di ML . Questo porta, quindi, a maggiore **spiegabilità** dei modelli di ML, migliore **generalizzazione e capacità di ragionamento** e sistemi di AI più **robusti e flessibili**, in grado di affrontare problemi complessi del mondo reale.

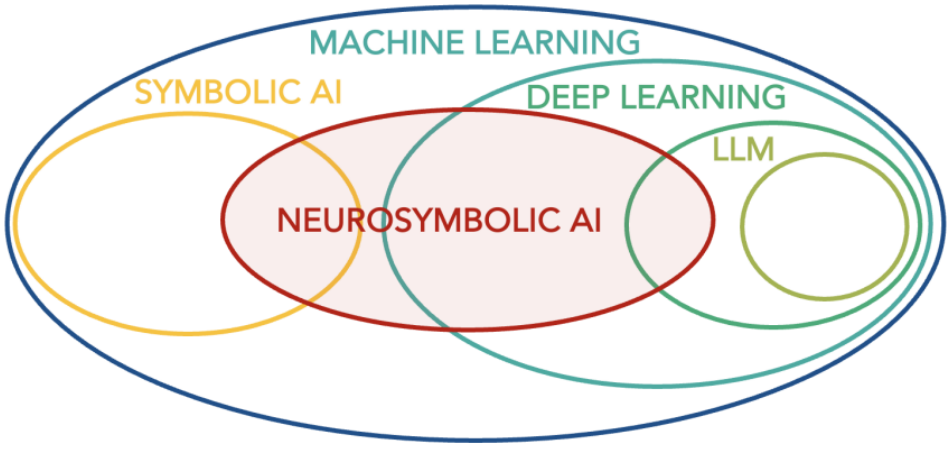


Figura 2.5: AI Neurosimbolica

Questa si divide in due parti fondamentali, che sono la Symbolic Knowledge Injection (SKI) e Symbolic Knowledge Extraction (SKE) (figura 2.6).

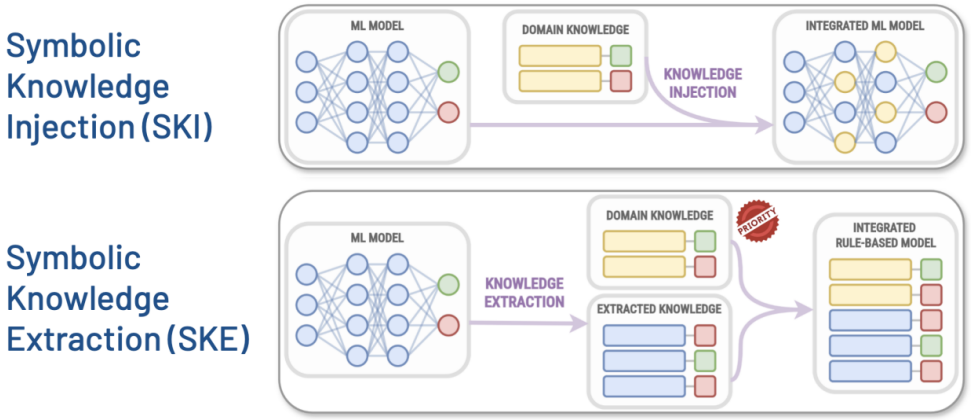


Figura 2.6: Symbolic Knowledje Injection ed Extraction

2.3.1 Symbolic Knowledge Injection

SKI si riferisce a qualsiasi procedura algoritmica che influenza il modo in cui i modelli effettuano inferenze, garantendo che le previsioni siano calcolate in funzione di una conoscenza simbolica o coerenti con essa.

Gli algoritmi SKI accettano come input una conoscenza simbolica fornita dall'utente e producono come output predittori di ML. Poiché questa conoscenza deve essere sia interpretabile dall'essere umano, che manipolabile algoritmicamente, è necessario che sia anche comprensibile dalla macchina.

Quale conoscenza possiamo integrare

Le regole logiche che possiamo incorporare nei modelli di ML attraverso metodi di SKI appartengono a diverse categorie.

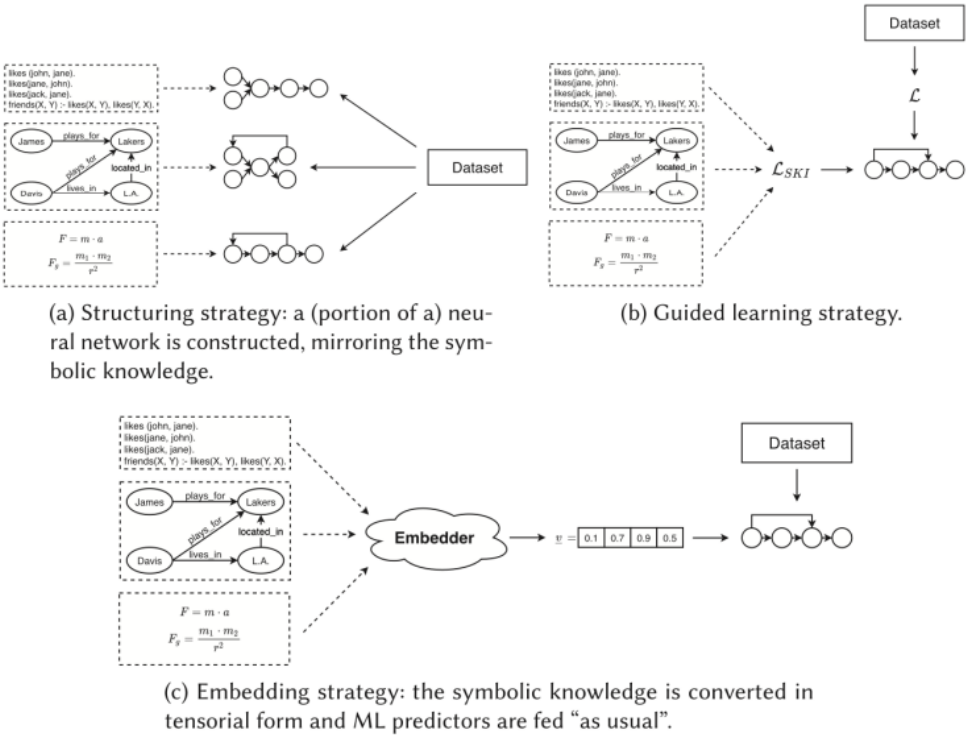
La **logica del primo ordine** (Full First-Order Logic) consente di esprimere definizioni e relazioni complesse attraverso termini ricorsivi, variabili, predicati di qualsiasi arietà e connettivi logici. Un caso specifico è la **logica di Horn**, nello stile di Prolog, che si basa su regole testa-corpo con predicati e termini, risultando particolarmente efficiente per il ragionamento basato su regole. **Datalog** rappresenta un sottinsieme della logica di Horn, caratterizzato dall'assenza di termini ricorsivi e dall'uso esclusivo di costanti o variabili, ed è impiegato principalmente per interrogazioni su basi di dati. Le **logiche modali**, invece, estendono le logiche precedenti introducendo operatori modali, rendendole adatte a rappresentare concetti come il tempo, come avviene nella logica temporale.

Un altro strumento di integrazione è costituito dai **grafi di conoscenza**, che

applicano logiche descrittive per rappresentare entità e relazioni. Questa tecnica è ampiamente utilizzata nel web semantico e nei sistemi di AI.

Infine, la **logica proposizionale** si basa su variabili booleane e connettivi logici, risultando particolarmente adatta per compiti di ragionamento binario più semplici.

Come funziona?



(a) Structuring strategy: a (portion of a) neural network is constructed, mirroring the symbolic knowledge.

(b) Guided learning strategy.

(c) Embedding strategy: the symbolic knowledge is converted in tensorial form and ML predictors are fed "as usual".

Figura 2.7: Strategie di Symbolic Knowledge Injection [16]

Questa conoscenza, come visto in precedenza, viene integrata seguendo diverse strategie. In figura 2.7 vediamo metodi come la **strategia di strutturazione** (*Structuring strategy*) in cui una parte della rete neurale viene costruita rispecchiando la conoscenza simbolica; la **strategia di apprendimento guidato** (*Guided learning strategy*) in cui la conoscenza simbolica guida il processo di apprendimento influenzando la funzione di perdita; la **strategia di incorporamento** (*Embedding strategy*) in cui la conoscenza simbolica viene convertita in una forma tensoriale e utilizzata come input per i predittori di ML.

Con quali modelli di ML possiamo usare SKI?

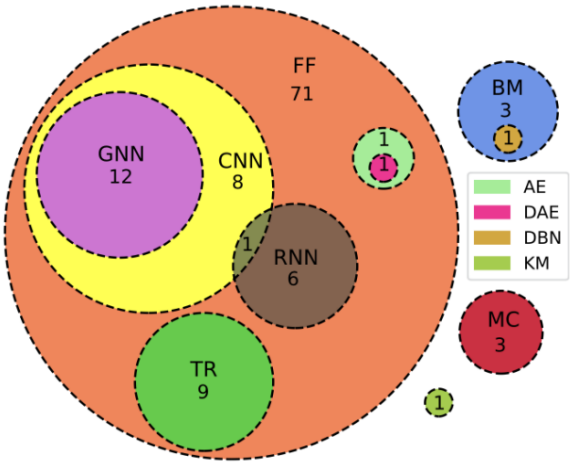


Figura 2.8: Metodi di ML su cui è possibile applicare metodi di SKI [16]

In figura 2.8 vediamo un diagramma di Venn che classifica i metodi SKI in base al tipo di predittore di ML a cui possono essere applicati. Nella tabella 2.1 sotto vediamo tutte le categorie presenti.

Tipo di rete	**Numero di metodi**
Reti neurali feed-forward	71
Reti convoluzionali	8
Reti neurali ricorrenti	6
Reti neurali per grafi	12
Transformer	9
Macchine di Boltzmann	3
Catene di Markov	3
Autoencoder	1
Deep belief networks	1
Denoising autoencoders	1
Kernel machines	1

Tabella 2.1: Distribuzione dei metodi nei diversi tipi di reti

2.3.2 Symbolic Knowledge Extraction

SKE si riferisce a qualsiasi procedura algoritmica che prende come input un modello di ML già addestrato e produce conoscenza simbolica come output. L'obiettivo è che la conoscenza estratta rifletta fedelmente il comportamento del predittore.

Il concetto di traslucenza

La traslucenza implica la necessità che questi metodi ispezionino la struttura interna del modello black-box sottostante durante l'estrazione delle regole. Ciò consente di comprendere meglio il funzionamento del modello e di rendere le sue decisioni più interpretabili.

I metodi di SKE garantiscono translucenza tramite metodi **decomposizionali** o **pedagogici**.

I metodi **decomposizionali** richiedono l'ispezione dei parametri interni del modello black-box ed estraggono conoscenza simbolica analizzando direttamente i suoi componenti.

I metodi **pedagogici**, invece, non necessitano di accesso ai parametri interni, ma si basano esclusivamente sugli output del modello per estrarre conoscenza simbolica. Questi ultimi spesso addestrano un modello surrogato interpretabile per rappresentare il comportamento del modello originale.

2.3.3 Train-Extract-Fix-Inject

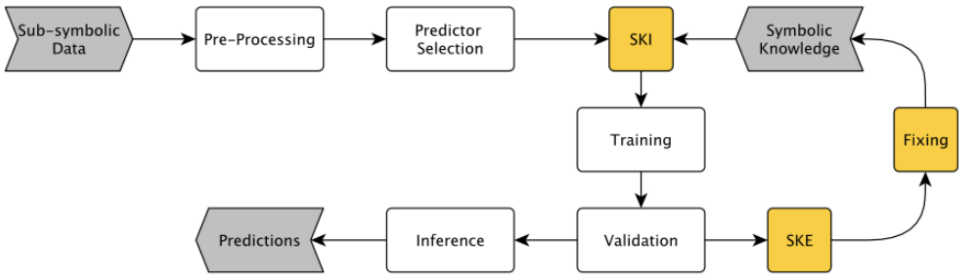


Figura 2.9: Flusso di apprendimento automatico arricchito conSKIe SKE [16]

La figura 2.9 illustra come le fasi di SKIe SKE si integrano nel flusso standard del ML. In particolare, viene evidenziato il ciclo train-extract-fix-inject (TEFI), un approccio iterativo che consente di migliorare i modelli di ML attraverso un processo ripetuto di incorporazione ed estrazione di conoscenza simbolica.

Come mostrato nella figura, SKI viene applicato prima o durante la fase di addestramento, permettendo di iniettare conoscenza simbolica nel modello. Questo può essere particolarmente utile per guidare l'apprendimento e migliorare la generalizzazione del modello. D'altra parte, SKE interviene dopo la fase di addestramento, consentendo di estrarre conoscenza simbolica dal modello addestrato. Questa conoscenza estratta può essere utilizzata per interpretare il comportamento del modello, identificare potenziali problemi e migliorare ulteriormente le sue prestazioni.

Il ciclo TEFI rappresenta un cambiamento significativo rispetto ai flussi di lavoro ML tradizionali, che tendono ad essere lineari. Questo ciclo iterativo consente di migliorare ripetutamente le performances e l'affidabilità dei modelli. Questo metodo, però, non è stato esplorato nel corso di questo studio, che si concentra esclusivamente sull'aspetto di iniezione della conoscenza.

2.4 Cenni storici

L'integrazione tra AI simbolica e ML ha radici profonde nella storia dell'informatica. Negli anni '80 e '90, l'attenzione si concentrava sui sistemi esperti, che utilizzavano regole simboliche definite da esperti umani per risolvere problemi complessi. Tra i pionieri di questi sistemi si ricordano Edward Feigenbaum definito il padre dei sistemi esperti, ponendo le basi per sistemi come MYCIN sviluppato da Edward H. Shortliffe [17]. Inoltre, ricordiamo John McCarthy considerato uno dei padri fondatori dell'AI simbolica. Tuttavia, questi sistemi si rivelarono limitati nell'adattarsi a nuovi contesti o nell'elaborare grandi quantità di dati.

Con l'avvento del deep learning nei primi anni 2010, l'interesse si è spostato verso approcci data-driven, capaci di apprendere automaticamente dai dati. Geoffrey Hinton, Yann LeCun e Yoshua Bengio sono stati tra i principali artefici della rivoluzione del deep learning, sviluppando architetture che hanno ridefinito il panorama dell'AI. Tuttavia, le limitazioni di questi modelli nel garantire trasparenza, interpretabilità e la necessità di grandi quantità di dati per il loro addestramento hanno riacceso l'interesse per approcci ibridi tra sistemi simbolici e statistici.

I metodi di SKI e SKE sono emersi come strategie per combinare il meglio dei due approcci: la precisione e interpretabilità di sistemi esperti e la flessibilità e la generalizzazione di sistemi statistici. Tra i principali approcci di SKI e SKE troviamo il Knowledge-Based Neural Networks (KBANN) [18], sviluppato da Richard Maclin e Jude Shavlik, che integra conoscenze simboliche come regole logiche nelle reti neurali; il Knowledge Injection via Network Structuring (KINS) [23], introdotto per adattare l'architettura delle reti neurali in

base a vincoli simbolici; il Logic Tensor Networks (LTN) [24], ideato da Luciano Serafini e Artur d'Avila Garcez, che combina logica simbolica e modelli tensoriali per l'apprendimento e il ragionamento.

2.5 Obiettivi

L'applicazione di SKI nel flusso di lavoro ML offre un approccio più flessibile e potente per lo sviluppo di modelli di apprendimento automatico. Questo approccio consente di combinare i vantaggi dell'apprendimento automatico con la capacità di incorporare e sfruttare la conoscenza simbolica, portando a modelli più accurati, interpretabili e affidabili.

Risulta quindi fondamentale approfondire queste tecniche e valutarne l'efficacia in questi contesti. Queste tecnologie possono, tramite l'integrazione di regole simboliche derivate da linee guida mediche ed esperti, consentire ai modelli di effettuare più accuratamente le proprie previsioni. Ad esempio prendendo come riferimento un dataset per la diagnosi del diabete, possiamo includere conoscenza nota che porta ad identificare un paziente come diabetico, come la presenza simultanea di iperglicemia e obesità, anche quando tali combinazioni sono sottorappresentate nei dati. Questo approccio migliora la capacità del modello di generalizzare su casi rari e fornisce un supporto decisionale più interpretabile per i professionisti sanitari.

Questo studio vuole, quindi, **esplorare i migliori metodi di SKI** presenti nella letteratura, con il fine di valutare la loro capacità nel migliorare le **performance** dei modelli in condizioni di dataset rumorosi, ridotti e sottorappresentati o di valutare come le performance di addestramento migliorano in termini di **tempi di convergenza**.

Inoltre, si pone come obiettivo quello di creare e testare **nuovi approcci** di SKI, valutandone l'efficacia in diversi scenari e applicazioni in ambito sanitario. Questi nuovi metodi verranno confrontati con quelli esistenti per analizzare i potenziali vantaggi in termini di interpretabilità, robustezza e generalizzazione del modello. Infatti, oltre alle performance di addestramento ed accuratezza si valuterà anche come questi metodi influenzano la **robustezza** dei modelli, osservando le performance in presenza di perturbazioni dei dati di vario genere.

Capitolo 3

Metodi

In questo capitolo vengono presentati i dati, le conoscenze simboliche, le metriche utilizzate per valutare le performance dei modelli e il design sperimentale. Vengono, inoltre, descritti i metodi di SKI già presenti in letteratura, e introdotti i nuovi metodi di perturbazione e di iniezione.

3.1 Dataset

3.1.1 Pima Indians Diabetes

Il dataset **Pima Indians Diabetes** [25] fornito dal *National Institute of Diabetes and Digestive and Kidney Diseases* e ha come obiettivo la previsione dell'insorgenza di diabete in base ad una serie di parametri diagnostici. Questo è composto da 768 elementi ed è stato selezionato da un database più ampio, applicando criteri che limitano l'analisi a pazienti donne di origine Pima, tutte di età pari o superiore a 21 anni. Le feature sono descritte in tabella 3.1. Il target è rappresentato da 'Outcome', che distingue tra pazienti malati (classe 1) e pazienti sani (classe 0).

Feature	Abbr.	Descrizione
Numero di gravidanze	NG	Numero di gravidanze della paziente.
Glucosio	GLC	Glucosio nel plasma a 2 ore
Pressione	BLP	Pressione sanguigna diastolica mmHg
Indice di Massa Corporea	BMI	Rapporto peso-altezza.
Livello di insulina	INS	Insulina nel sangue (µU/mL).
Spessore piega cutanea	SPC	Spessore del tricipite (mm).
Età	AGE	Età della paziente (anni).
Diabetes Pedigree Function	DPF	Predisposizione al diabete familiare.

Tabella 3.1: Descrizione delle feature del dataset

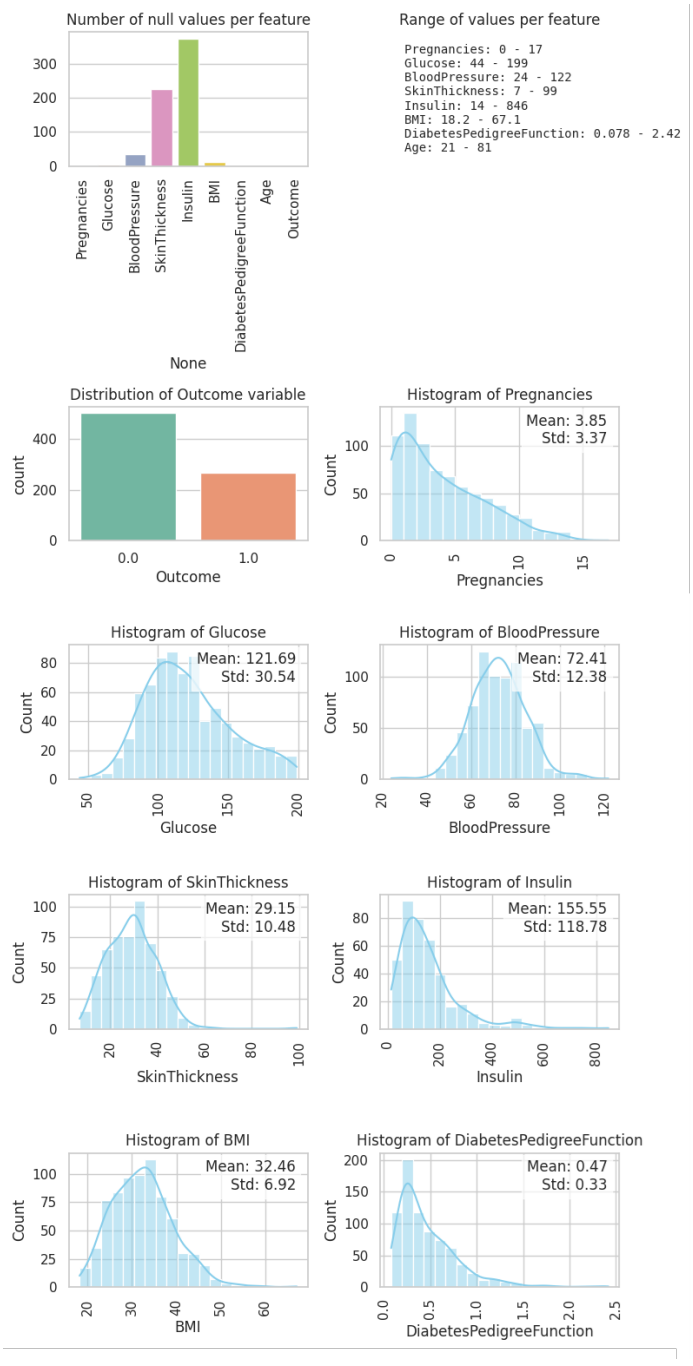


Figura 3.1: Nell'immagine una serie di grafici che analizzano il dataset Pima. Il primo grafico mostra le feature per cui sono stati trovati elementi nulli. Il secondo ci mostra il range per ogni feature. Il terzo ci mostra il numero di elementi per ogni classe del dataset. Tutti gli altri ci mostrano la distribuzione dei valori di ogni feature del dataset.

3.1.2 Breast Cancer Wisconsin (Original)

Il dataset **Breast Cancer Wisconsin (Original)** [29] è un insieme di dati diagnostici utilizzato per la classificazione di tumori al seno come benigni o maligni. Comprende 699 campioni con 9 feature e un target, raccolti attraverso l'analisi di immagini digitali.

Il dataset è stato creato da *Dr. William H. Wolberg* dell'Università del Wisconsin-Madison. I dati sono stati raccolti tra il 1989 e il 1991 e successivamente resi pubblici per la ricerca sul machine learning attraverso l'*UCI Machine Learning Repository*.

Feature	Abbr.	Descrizione
Clump Thickness	CT	Spessore medio cellule epiteliali aggregate.
Uniformity of Cell Size	UCS	Uniformità della dimensione cellulare.
Uniformity of Cell Shape	UCH	Regolarità della forma cellulare.
Marginal Adhesion	MA	Capacità adesione delle cellule ai margini.
Single Epithelial Cell Size	SECS	Dimensione delle cellule epiteliali isolate.
Bare Nuclei	BN	N. nuclei non circondati da citoplasma.
Bland Chromatin	BC	Uniformità della cromatina nel nucleo.
Normal Nucleoli	NN	Numero di nucleoli visibili nel nucleo.
Mitoses	M	Frequenza delle mitosi osservate.
Class	C	Diagnosi: benigno (2) o maligno (4).

Tabella 3.2: Descrizione delle feature del dataset

Nella figura 3.2 visualizziamo diversi grafici che danno informazioni più dettagliate sul dataset per visualizzarne, la distribuzione delle features, valori nulli e sbilanciamento del dataset.

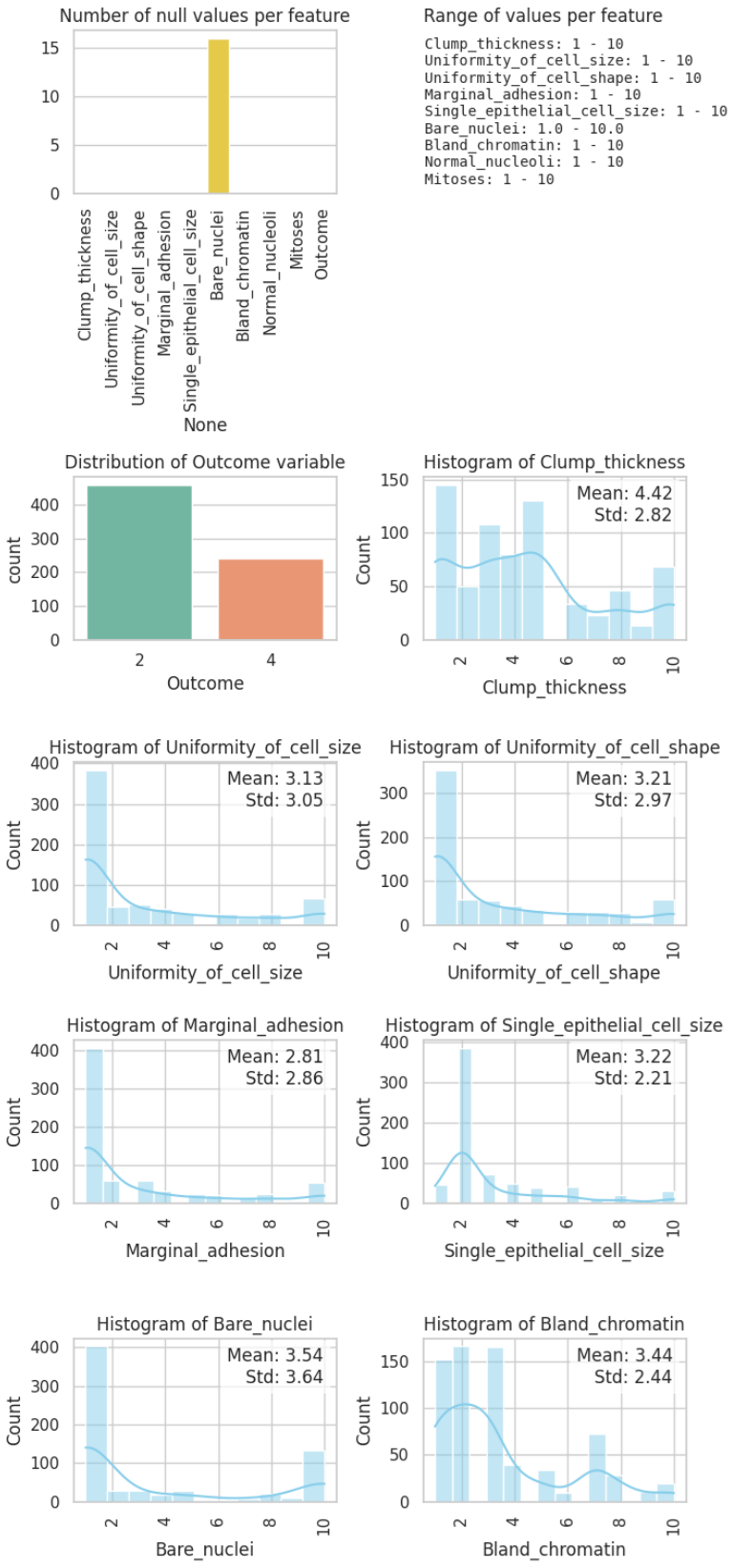


Figura 3.2: Analisi Breast Cancer Dataset

3.2 Conoscenze

Le conoscenze utilizzate per educare i modelli, nel corso di questa sperimentazione, sono tre, di cui una per il dataset Pima e due per il dataset Breast Cancer Wisconsin.

La conoscenza utilizzata su Pima è descritta dalla tabella che segue. Per una migliore comprensione di quanto riportato si può fare riferimento alla tabella 3.1 che descrive le feature su cui la conoscenza si basa.

Regola	Outcome
$GLC > 126 \wedge BMI > 30$	Diabetico
$GLC <= 100 \wedge BMI <= 30$	Non diabetico

Tabella 3.3: Conoscenza del dataset Pima [26]

La conoscenza utilizzata su Breast Cancer Wisconsin è descritta dalle tabelle che seguono. Per una migliore comprensione di quanto riportato si può fare riferimento alla tabella 3.2 che descrive le feature su cui la conoscenza si basa. Le conoscenze sono due, tra cui distinguiamo in ordine, quella del 2001 di **Duch, Adamczak e Grabczewski (DAG)** [27] e quella del 2015 di **Hayashi e Nakano (HN)** [28].

Regola	Outcome
$BN > 5.5$	Maligno
$CT > 6.5$	Maligno
$BN <= 5.5 \wedge CT <= 6.5$	Benigno

Tabella 3.4: Conoscenza del dataset Breast Cancer Wisconsin di DAG

Regola	Outcome
$BN <= 1.5$	Benigno
$BN > 1.5 \wedge CT > 4.5$	Maligno
$BN > 6.5 \wedge CT <= 4.5$	Maligno
$1.5 < BN <= 6.5 \wedge CT <= 4.5$	Benigno

Tabella 3.5: Conoscenza del dataset Breast Cancer Wisconsin di HN

3.3 Metriche

Le metriche utilizzate per determinare le performance dei modelli sono diverse. Sono state considerate le metriche classiche come l'accuracy che indica la percentuale di elementi predetti in maniera corretta sull'intero target. Sono state prese in considerazione anche metriche specifiche per dataset sbilanciati e per classificazione binaria.

Poichè nel nostro caso, la classe con meno elementi è proprio la classe 1, ed è anche quella su cui vogliamo porre maggiore attenzione, considereremo una metrica chiamata *recall*. Questa metrica esprime la percentuale degli elementi della classe 1 predetti correttamente. La recall viene calcolata come il rapporto tra i veri positivi e la somma tra il numero dei veri positivi e il numero di falsi negativi. Indichiamo con TP, TN, FP, FN rispettivamente i veri positivi, veri negativi, falsi positivi, falsi negativi.

$$Recall = \frac{TP}{(TP + FN)}$$

Allo stesso modo terremo traccia dell'accuratezza sugli elementi con target zero. Questa metrica, corrispettiva della recall per la classe zero e viene chiamata specificity.

$$Specificity = \frac{TN}{(TN + FP)}$$

Abbiamo, poi, la F1-score che consiste nella media armonica tra la precision (numero di veri positivi diviso il numero di tutti i risultati positivi) e la recall.

$$F1 = 2 \cdot \frac{precision \cdot recall}{(precision + recall)}$$

La recall però non tiene in considerazioni aspetti come lo sbilanciamento, per questo affianchiamo alle metriche precedenti la balanced accuracy. Questa metrica è la media tra la recall così come l'abbiamo vista (in altri casi detta sensitivity) e la specificity.

$$Balanced\_accuracy = \frac{Sensitivity + Specificity}{2}$$

Infine come ultime due metriche valuteremo l'accuratezza dei modelli rispetto agli elementi della conoscenza con target zero e elementi della conoscenza con target uno. Le chiameremo rispettivamente *knowledge_0* e *knowledge_1* e saranno calcolati rispettivamente come la specificity e la recall sugli elementi della conoscenza.

3.4 Metodi di Symbolic Knowledge Injection

I metodi di SKI esistenti analizzati in questo studio, sono tre: Knowledge Based Neural Network, Knowledge Injection via Network Structuring, Logic Tensor Network.

3.4.1 Knowledge Based Neural Network

KBANN [18] è un metodo in grado di costruire una rete neurale partendo da un set di regole simboliche, come base strutturale per i pesi e le connessioni della rete. Il processo è strutturato in due fasi principali: la **costruzione iniziale della rete** e la **fase di addestramento**.

Costruzione iniziale della rete

Le regole simboliche vengono trasformate in connessioni e pesi iniziali. Le disgiunzioni vengono prima trasformate in un insieme di regole per cui ognuno ha un solo antecedente, come mostrato in figura 3.3.

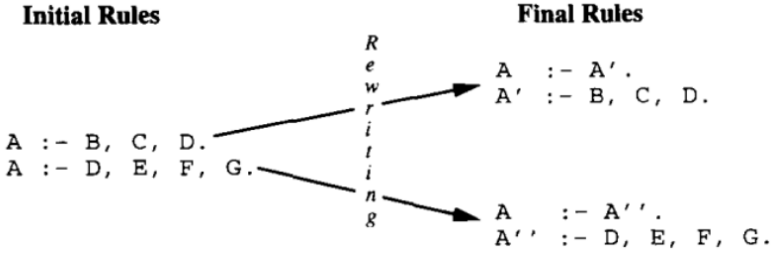


Figura 3.3: Trasformazione delle regole in KBANN

Questo facilita la trasformazione delle regole in strutture neurali, dato che in questo modo ogni neurone dovrà gestire solo un insieme con esclusivamente congiunzioni o disgiunzioni, dividendo la gestione composta di queste in più livelli.

La trasformazione delle regole avviene secondo la tabella 3.6:

Knowledge base		Neural network
Final conclusions	⇔	Output units
Supporting facts	⇔	Input units
Intermediate Conclusions	⇔	Hidden units
Dependencies	⇔	Weighted connections

Tabella 3.6: Mapping tra Knowledge base e Neural network [18]

Successivamente KBANN attribuisce un livello numerico a ciascuna unità nella rete KBANN. Sebbene questo numero non abbia un'utilità diretta, è fondamentale per poter eseguire i passaggi successivi. Il livello di un'unità viene determinato calcolando la lunghezza del percorso più lungo che la collega a un'unità di input.

Vengono poi aggiunte unità nascoste alla rete risultante per permettere alla rete di apprendere caratteristiche derivate non esplicitate nel set di regole iniziale. Questo passaggio è facoltativo, poiché le regole di base fornite sono spesso sufficienti e non richiedono nuove unità nascoste.

L'algoritmo crea collegamenti nella rete con peso iniziale pari a zero, basandosi sulla numerazione delle unità definita prima. Vengono aggiunti collegamenti che connettono ciascuna unità con livello n-1 alle unità con livello n. L'ultimo passaggio nella conversione delle regole prevede la perturbazione di tutti i pesi della rete, aggiungendo a ciascun peso un piccolo numero casuale. Questa modifica è così piccola da non alterare le operazioni della rete, ma è sufficiente a evitare eventuali problemi derivanti dalla simmetria, una situazione in cui neuroni all'interno dello stesso strato della rete finiscono per apprendere esattamente le stesse funzioni, ostacolando il processo di apprendimento e impedendo alla rete di esplorare diverse soluzioni al problema.

Fase di addestramento

Una volta impostata la rete con la conoscenza simbolica, il modello viene addestrato su dati empirici. Durante questa fase, i pesi vengono raffinati attraverso l'uso di algoritmi di apprendimento automatico, adattando la rete ai dati senza però stravolgere la conoscenza inizialmente iniettata.

3.4.2 Knowledge Injection via Network Structuring

KINS [23] utilizza una diversa strategia per l'iniezione di conoscenza, incentrata sulla modifica della struttura della rete iniziale affinché rifletta direttamente le relazioni simboliche e logiche contenute nella conoscenza. KINS codifica regole logiche in funzioni a valori reali, che vengono poi rappresentate tramite dei neuroni che affiancano la rete di partenza su cui viene applicato l'algoritmo. La conoscenza viene fornita come clausole di Horn della forma $\varphi \leftarrow \psi_1 \wedge \psi_2 \wedge ... \psi_n$.

Per trasformare la conoscenza, KINS si avvale di una logica ispirata a quella di Łukasiewicz riportata nella tabella in figura 3.4:

Formula	C. interpretation	Formula	C. interpretation
$[\![\neg\phi]\!]$	$\eta(1-[\![\phi]\!])$	$[\![\phi \leq \psi]\!]$	$\eta(1+[\![\psi]\!]-[\![\phi]\!])$
$[\![\phi \wedge \psi]\!]$	$\eta(min([\![\phi]\!],[\![\psi]\!]))$	$[\![class(\bar{X},\mathtt{y}_i) \leftarrow \psi]\!]$	$[\![\psi]\!]^*$
$[\![\phi \vee \psi]\!]$	$\eta(max([\![\phi]\!],[\![\psi]\!]))$	$[\![\mathsf{expr}(\bar{X})]\!]$	$\mathsf{expr}([\![\bar{X}]\!])$
$[\![\phi = \psi]\!]$	$\eta([\![\neg(\phi \neq \psi)]\!])$	$[\![\mathtt{true}]\!]$	1
$[\![\phi \neq \psi]\!]$	$\eta(\lvert[\![\phi]\!]-[\![\psi]\!]\rvert)$	$[\![\mathtt{false}]\!]$	0
$[\![\phi > \psi]\!]$	$\eta(max(0,\frac{1}{2}+[\![\phi]\!]-[\![\psi]\!]))$	$[\![X]\!]$	x
$[\![\phi \geq \psi]\!]$	$\eta(1+[\![\phi]\!]-[\![\psi]\!])$	$[\![\mathtt{k}]\!]$	k
$[\![\phi < \psi]\!]$	$\eta(max(0,\frac{1}{2}+[\![\psi]\!]-[\![\phi]\!]))$	$[\![p(\bar{X})]\!]^{**}$	$[\![\psi_1 \vee \ldots \vee \psi_k]\!]$

Figura 3.4: Logica di Łukasiewicz

I valori scalari delle condizioni vengono poi trasformati in valori nell'intervallo [0,1] secondo la funzione $\eta : \mathbb{R} \to [0,1]$ in figura 3.5

$$\eta(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } 0 < x < 1 \\ 1 & \text{if } x \geq 1 \end{cases}$$

Figura 3.5: Funzione di trasformazione scalari in KINS

La conversione in nodi della rete neurale avviene come mostrato nella figura 3.6

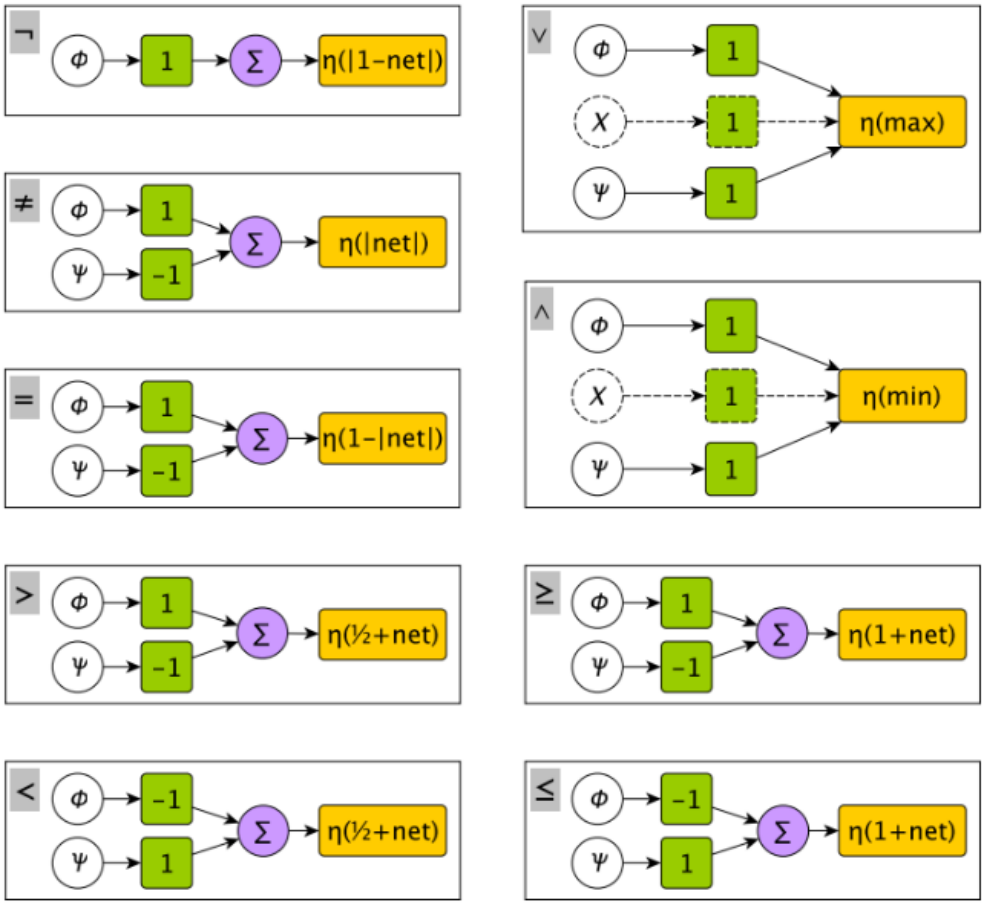


Figura 3.6: Conversione delle regole in nodi in KINS

I cerchi bianchi sono le variabili di input (I), le caselle verdi rappresentano i pesi corrispondenti (W), i cerchi viola sono la somma degli input pesati (WI). I rettangoli gialli sono le funzioni di attivazione, net è l'output di WI, max e min sono rispettivamente il massimo e il minimo dei valori di input, η è la funzione descritta precedentemente.

KINS può essere applicato sia a reti non addestrate, sia a reti parzialmente addestrate. Richiede, però, che la rete abbia un layer di input e di output, e venga addestrata tramite gradient descent o algoritmi simili.

3.4.3 Logic Tensor Network

In LTN [24], la rete viene addestrata come una normale rete neurale tramite Stochastic Gradient Descent (SGD) e backpropagation. Tuttavia, ciò che distingue il loro addestramento è la presenza di vincoli logici che condizionano l'ottimizzazione, rappresentando la conoscenza simbolica in una funzione

reale che quantifica il grado di conformità ai vincoli simbolici. Le regole da introdurre in LTN sono espresse in logica del primo ordine e possono includere predicati, funzioni, connettivi logici e quantificatori.

Una fase fondamentale del processo di traduzione è il grounding: una funzione G che assegna significati concreti e numerici agli elementi logici in modo che possano essere gestiti dal modello attraverso una rappresentazione differenziabile.

Grounding delle Costanti e delle Variabili: Ogni costante o variabile, che rappresenta oggetti specifici del dominio, viene associata a tensori reali, che ne descrivono le caratteristiche o le "feature" numeriche. Alle variabili è associata una sequenza di tensori. Ad esempio, una costante "Alice" che rappresenta una persona potrebbe essere mappata su un vettore che rappresenta le sue caratteristiche (età, altezza, ecc.).

Grounding delle Funzioni e dei Predicati: A funzioni sono associati a operazioni su tensori che restituiscono altri tensori. I predicati, invece, producono valori di verità tra 0 e 1, dove 0 indica una falsità assoluta e 1 una verità assoluta.

Connettivi e quantificatori: La congiunzione ($\wedge$), la disgiunzione ($\vee$), l'implicazione ($\rightarrow$) e la negazione ($\neg$) sono associate, rispettivamente, a una t-norm, una t-conorm, implicazione fuzzy e negazione fuzzy. Questi sono concetti chiave della logica fuzzy, che estende la logica classica permettendo valori di verità compresi tra 0 e 1. Dove a e b sono il grado di verità delle affermazioni, definiamo di seguito questi concetti.

La t-norm è una norma triangolare con cui viene approssimato il connettore logico "e". Una delle forme più comuni è:

$$T(a, b) = min(a, b)$$

La t-conorm è una norma triangolare con cui viene approssimato il connettore logico "o". Una delle forme più comuni è:

$$T(a, b) = max(a, b)$$

La negazione fuzzy generalizza il “non A” ed è spesso definita come:

$$N(a) = 1 - a$$

L'implicazione fuzzy rappresenta la relazione "se A allora B", dove una delle formulazioni più comuni, derivante dall'equivalenza logica per cui $a \rightarrow b \Leftrightarrow$

$\neg a \vee b$, è la seguente:

$$I(a, b) = \max(1 - a, b)$$

Entrambe permettono di modellare in modo più flessibile ragionamenti incerti o graduali. I quantificatori ($\forall$, $\exists$) sono implementati come metodi di aggregazione differenziabile.

La figura 3.7 mostra come la query simbolica "tutti hanno un amico italiano" $(\forall x \exists y \mid R(x, y) \wedge A(y))$, dove R è la relazione x è amico di y e A il predicato per cui y è italiano, viene trasformata in tensori dalla funzione di grounding.

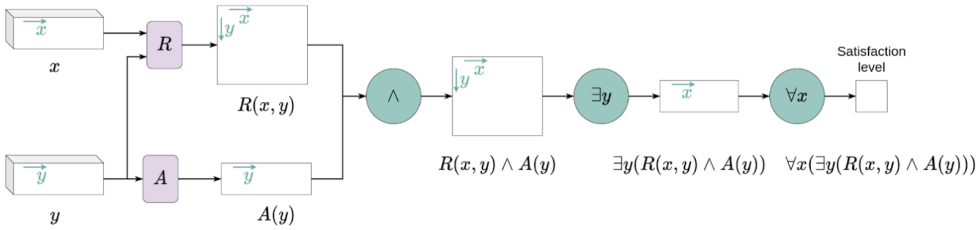


Figura 3.7: Esempio query con LTN

I predicati potrebbero essere anche implementati attraverso reti neurali che hanno appreso specifiche caratteristiche aiutando a ricavare la veridicità di un predicato come visibile in figura 3.8. In questo caso, la veridicità di un predicato, viene prevista attraverso un'apposita rete neurale, e il suo output viene utilizzato come input dei connettori e quantificatori successivi.

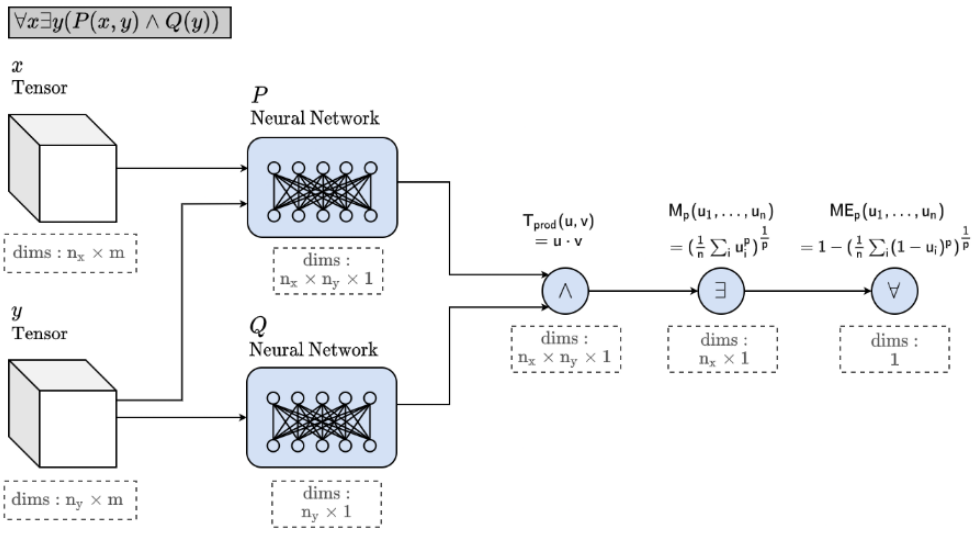


Figura 3.8: Esempio query con rete neurale con LTN

Questi Tensor Computational Graph, vengono poi usati per la valutazione dell'errore della rete in base a quanto questa soddisfa le regole logiche descritte. In base a questa loss e tramite la backpropagation il modello riesce, poi, ad apprendere dai dati, tenendo conto anche delle regole imposte.

3.5 Nuovi metodi

3.5.1 Siamese Training Injection

Nel corso di questo studio è stato proposto e testato un nuovo metodo di SKI chiamato **Siamese Training Injection (STI)**. Questo è un metodo di addestramento che permette di addestrare una rete neurale contemporaneamente sulla conoscenza in modo da rispettarne i vincoli e sui dati per apprendere nuove relazioni.

Per elementi appartenenti alla conoscenza, intendiamo tutti quei dati che soddisfano una o più regole della conoscenza. Ad esempio, per la regola $Glucose > 126, BMI > 30 \rightarrow 1$ considereremo appartenente alla conoscenza tutti quegli elementi con glucosio maggiore di 126, con BMI maggiore di 30 e con output 1. Risulta importante considerare anche l'output, poiché possono essere presenti elementi con output opposto rispetto a quello indicato dalla regola, compromettendo la conformità della rete alla conoscenza.

Il funzionamento di questo metodo si basa sul concetto di rete siamese, che permette di addestrare una rete contemporaneamente su più input e output diversi. Il modello di base viene concatenato in parallelo a se stesso n volte, ottenendo un modello più grande che prende n input e restituisce n output. Dove per input si intende un insieme di m feature e per output un valore numerico che classifica l'input ricevuto.

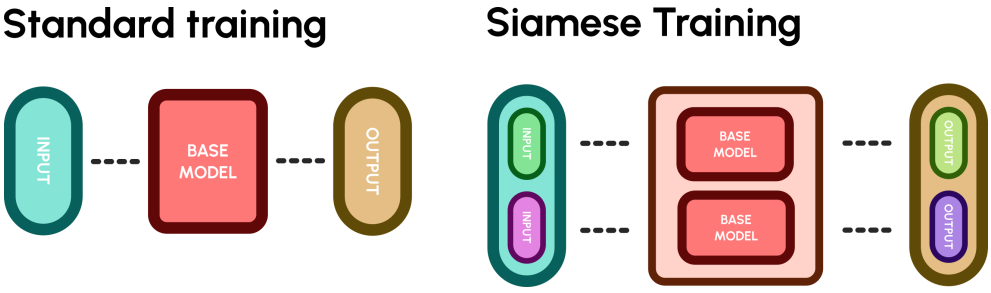


Figura 3.9: Differenza tra addestramento standard e siamese

Questo metodo di addestramento viene utilizzato nel contrastive learning, che è una tecnica che permette al modello di apprendere le caratteristiche che

accomunano esempi simili e li distinguono da esempi dissimili. L'obiettivo è ridurre la distanza tra coppie simili e aumentare quella tra coppie dissimili nello spazio delle rappresentazioni apprese. Le reti siamesi hanno un ruolo fondamentale in questo contesto, permettendo di presentare allo stesso modello due esempi differenti, e tramite una appropriata funzione di costo si riesce a modificare appropriatamente la distanza tra questi esempi nello spazio di output [2] [3].

A differenza di una rete con la stessa architettura, costruita senza replicare lo stesso modello di base, in questo caso ci sono parti della rete che condividono i pesi.

Questo è reso possibile grazie alla struttura computazionale di TensorFlow (o altre librerie) e al meccanismo della backpropagation attraverso grafi computazionali. TensorFlow costruisce un grafo computazionale dinamico che tiene traccia delle operazioni eseguite e dei tensori coinvolti. Quando si calcola la loss, TensorFlow esegue la backpropagation seguendo questi passaggi:

1. **Calcolo del gradiente**: Viene calcolato il gradiente della loss rispetto a ciascun nodo del grafo (operazioni e pesi) usando la regola della catena.
2. **Accumulo del gradiente**: I gradienti derivati da ogni percorso vengono sommati per ogni parametro condiviso.
3. **Aggiornamento dei pesi**: TensorFlow aggiorna i pesi in base al gradiente complessivo.

Gli input e gli output per il metodo STI possono essere scelti in vario modo. Possiamo distinguere due variabili principali rispetto alle quali possiamo caratterizzare STI: numero di parallelizzazioni e la tipologia di input.

I test per numero di parallelizzazioni effettuati sono tre. Nel caso di **due parallelizzazioni**, gli input sono divisi tra elementi appartenenti alla conoscenza ed elementi non appartenenti. La rete viene quindi addestrata contemporaneamente su entrambi i vincoli, rispettando la conoscenza e generalizzando. Abbiamo poi **tre parallelizzazioni**, dove in una sono i dati non appartenenti alla conoscenza, in una quelli appartenenti alla conoscenza con output uno e l'ultima con elementi della conoscenza con output zero. Nel caso di **quattro parallelizzazioni**, gli input sono divisi tra elementi appartenenti alla classe zero ed elementi appartenenti alla classe uno, a loro volta divisi tra elementi della conoscenza ed elementi non appartenenti alla conoscenza. In questo modo si costringe la rete a rispettare tutti e quattro i vincoli contemporaneamente.

Possiamo quindi distinguere anche la tipologia di input. Gli **elementi della conoscenza ottenuti dai dati** sono quelli estratti dai dati attraverso maschere booleane. In questo caso, generalmente, gli elementi individuati dalla

conoscenza presenti nei dati sono meno di quelli non appartenenti alla conoscenza. Quindi gli elementi appartenenti possono essere replicati per accostare sempre un elemento della conoscenza ad un elemento dei dati. Successivamente, abbiamo gli **elementi della conoscenza generati** tramite un processo di data augmentation, seguendo precisi processi di generazione, che discutiamo di seguito.

La generazione di elementi avviene con un processo randomico guidato dalla conoscenza. Si generano n elementi per ogni regola della conoscenza, fino ad arrivare al numero di elementi desiderato. Le feature non interessate dalla conoscenza vengono generate randomicamente nell'intero range della feature. Mentre, le feature che sono condizionate dalla conoscenza vengono generate in un range che rispetta la regola di riferimento. Il target di ogni elemento generato viene indicato dalla regola stessa. Ad esempio, considerando la regola $Glucose > 126 \wedge BMI > 30 \rightarrow 1$ genereremo elementi che hanno tutte le feature selezionate randomicamente, mente per glucosio verrà selezionato un valore tra $(126, MassimoGlucosio]$ e per BMI verrà selezionato randomicamente un valore nell'intervallo $(30, MassimoBMI)$. Il target sarà impostato a 1.

Con lo scopo di migliorare la robustezza del metodo, il range per la generazione dei valori di feature non condizionati dalla conoscenza, verrà ristretto ad un intorno del valore medio.

Un'altra strada che verrà esplorata per STI, è la modifica della pipeline di addestramento, antecedendo una fase di preaddestramento sugli elementi della conoscenza, alla fase di addestramento siamese. L'idea è che questo metodo ci permette di trovare un minimo nella funzione di costo per la conoscenza, e poi muoverci successivamente nell'intorno di questo minimo (rimanendo vincolati in tutto il processo alla conoscenza stessa) per trovare le migliori performance generali, tenendo alte quelle della conoscenza.

Inoltre, essendo il metodo basato sui dati, un ulteriore miglioramento che possiamo apportare per migliorare la consistenza del modello iniettato con la conoscenza, è la pulizia del dataset degli elementi contrari alla conoscenza. Definiremo elementi contrari, tutti quegli elementi del dataset che hanno le feature che rispettano la conoscenza ma output opposto.

Infine per l'addestramento parallelo viene usata una generalizzazione della binary crossentropy, in cui si da un peso diverso ad ognuno degli output. Questo permette di dare un peso diverso alla conoscenza rispetto ai dati.

3.6 Perturbazioni

I metodi di SKI potrebbero anche migliorare la robustezza dei modelli iniettati durante previsioni su dati rumorosi e perturbati. A questo scopo andremo a testare le performance dei vari metodi a seguito di un addestramento dei modelli su dati perturbati secondo varie modalità.

La tecnica del **Label Flipping** prevede, in un contesto di classificazione binaria, l'inversione delle etichette di un certo numero di elementi scelti randomicamente. Ad esempio un dataset per la previsione di email "spam" o "non spam" potrebbe essere qualcosa come il contenuto della seguente tabella:

Email	Outcome
$Email1$	1
$Email2$	0
$Email3$	0
$Email4$	1

Tabella 3.7: Esempio dataset 1 non perturbato

A seguito della perturbazione con probabilità impostata al 50% otterremmo questo risultato:

Email	Outcome
$Email1$	1
$Email2$	0
$Email3$	1
$Email4$	0

Tabella 3.8: Esempio Label flipping

In cui la metà delle etichette sono state invertite.

Nel caso del **Feature Noise**, ai dati di ogni feature viene aggiunto un errore randomico distribuito secondo una gaussiana con media zero e deviazione standard pari a quella dei dati. La deviazione viene poi moltiplicata per l'entità della perturbazione da applicare, in modo che al crescere dell'entità abbiamo più probabilità di allontanarci dal valore iniziale. Per esempio, una entry di un dataset con due feature come nella tabella:

Feature 1	Feature 2	Outcome
50	4	1

Tabella 3.9: Esempio dataset 2 non perturbato

Con un'entità piccola potrebbe diventare come in tabella 3.10.

Feature 1	Feature 2	Outcome
56	3.8	1

Tabella 3.10: Esempio 1 Feature Noise

Con un'entità grande potrebbe diventare come in tabella 3.11.

Feature 1	Feature 2	Outcome
30	10	1

Tabella 3.11: Esempio 2 Feature Noise

Con la tecnica del **Data Dropping**, una certa percentuale degli elementi del dataset di training viene eliminata per testare come il modello si comporta in presenza di dati limitati.

Nell'esempio in tabella 3.8 applicando un data dropping con entità al 50% potremmo ottenere questo risultato:

Email	Outcome
*Email*1	1
*Email*4	1

Tabella 3.12: Esempio Data Dropping 50%

Infine, una nuova tecnica di perturbazione è stata testata: con la **Feature Corruption** una certa percentuale delle feature nei dati, scelte randomicamente, viene impostata a valori fuori dal normale range della relativa feature, simulando così una corruzione dei dati. Si dà, inoltre, la possibilità di impostare quanto la feature si allontanerà dal suo normale range. L'allontanamento viene calcolato in base alla distanza tra il valore minimo e il valore massimo della feature, che viene poi moltiplicato per l'entità della perturbazione e sommato alla feature.

Considerando il dataset in tabella 3.9 e considerando come range di riferimento [25,80] per la feature 1 e un range [0, 10] per la feature 2, mostreremo due casi di perturbazione del dataset.

Nella tabella 3.13 mostreremo il caso in cui l'entità del distacco è 1, e quini verrà sommato il valore $(feature\_max - feature\_min)$ all'estremo del range della feature di riferimento. Nella tabella, vediamo che per la feature 1 viene sommato in positivo: $(80 - 25) + 80$. Mentre per la feature 2 avremo $0 - (10 - 0)$. Infatti possiamo uscire fuori dal range della feature sia verso destra che versio sinistra.

Feature 1	Feature 2	Outcome
135	-10	1

Tabella 3.13: Esempio 1 Feature corruption

Aumentando l'entità del distacco, aumenteremo la distanza dal range normale, moltiplicando $(feature\_max - feature\_min)$ per l'entità della pertrbazione. Impostando a 5 l'entità possiamo ottenere il risultato in tabella 3.14

Feature 1	Feature 2	Outcome
355	-50	1

Tabella 3.14: Esempio 2 Feature corruption

Invece, la probabilità di perturbazione non farà altro che decidere a quante entry la perturbazione verrà applicata, in modo simile a quanto visto per il label flipping.

Studieremo, quindi, le performance dei modelli secondo le metriche sopra indicate (sezione 3.3) al variare dell'entità di queste perturbazioni per valutare come l'iniezione di conoscenza nei modelli ne migliori la robustezza.

3.7 Preprocessing

I dati sono stati opportunamente preprocessati prima di procedere con gli esperimenti, riempiendo eventuali dati mancanti o sostituendo quei dati che non descrivono la realtà in modo appropiato.

Il dataset Pima Indians, non presenta di per se dei valori nulli, ma ha comunque dei valori fisiologicamente implausibili, in quanto diverse feature (GLC, BLP, SPC, INS, BMI, DPF, AGE) presentano dei valori impostati a 0. Per

questo motivo, non essendo una possibile rappresentazione della realtà questi valori sono stati considerati nulli, producendo l'analisi visualizzata in figura 3.1. Successivamente, in fase di preprocessing, questi sono stati sostituiti con la mediana della rispettiva feature, in modo da ottenere una rappresentazione più realistica dei dati.

Invece, il dataset breast cancer wisonsin, presenta valori nulli per la feature BN. Come visto dall'analisi in figura 3.2 il valore più frequente per questa feature risulta lo 0. Per questo motivo tutti i valori nulli sono stati sostituiti con il valore 0.

3.8 Design sperimentale

Questa sezione vuole descrivere l'approccio adottato e le configurazioni sperimentali impiegate per condurre gli esperimenti. Inoltre, verranno discusse le ipotesi alla base dello studio e le strategie implementate per garantire la validità e la riproducibilità dei risultati.

Inizialmente, per garantire una generazione randomica controllata e ripetibile, tutti i seed delle librerie random, numpy e tensorflow, sono stati impostati a 42. Questa impostazione viene ripetuta ogni volta che si passa al metodo di SKI successivo.

Il modello di base non educato, identico in tutti gli esperimenti per garantirne la comparabilità diretta, è strutturato in modo preciso. Esso inizia con un layer composto da **64 unità** con funzione di attivazione *leaky relu*, seguito da un layer di **batch normalization**. Successivamente, è presente un secondo layer con **32 unità**, anch'esso dotato di attivazione *leaky relu*, cui segue un ulteriore layer di **batch normalization**. Infine, il modello si conclude con una singola unità di output che utilizza una funzione di attivazione **sigmoidale**.

Allo stesso scopo, il modello di base viene impostato con l'initializer "glorot uniform" con seed impostato a 42 ad ogni generazione del modello. Questo garantisce che le performance siano indipendenti dall'iterazione in cui il modello viene generato, inizializzandolo sempre con gli stessi pesi iniziali.

I parametri che caratterizzano l'addestramento sono stati definiti in modo da garantire coerenza e confrontabilità tra i diversi metodi. Ogni metodo utilizza il proprio *learning rate*: tutti sono impostati a 0.001, ad eccezione di KINS, per cui è stato scelto un valore pari a 0.01. L'addestramento prevede l'impiego dell'**early stopping**, con una pazienza di 20 epoche e il ripristino automatico dei pesi migliori ottenuti. Per tutti i metodi viene utilizzato l'**ottimizzatore** *Adam*, mentre la funzione di loss adottata è la classica *binary cross entropy*.

Le classi vengono bilanciate attraverso la funzione compute_class_weight del modulo class_weight del pacchetto utils della libreria sklearn, con il parametro class_weight impostato su "balanced".

I dati sono stati normalizzati tramite l'utilizzo di MinMaxScaler di sklearn, con un range tra 0 e 1.

Infine, l'addestramento è condotto per **100 epoche** con un **batch size** di 32 per tutti i metodi.

Per i metodi di perturbazione sono stati utilizzati nove livelli di crescente entità. Per label flipping e feature corruption i livelli di perturbazione applicati sono [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9], mente per feature noise i livelli sono : [1, 2, 3, 4, 5, 6, 7, 8, 9]. Per ogni livello di perturbazione l'addestramento viene ripetuto valutando le performance di ogni modello in relazione ad ogni livello di perturbazione.

La validazione dei modelli viene conseguita utilizzando la cross validation, dividendo il set dei dati in 4 folds. Questo permette di valutare le prestazioni del modello in modo più robusto, mitigando il rischio di overfitting e permette di ottenere una stima più affidabile delle prestazioni di generalizzazione del modello, poiché ogni osservazione viene utilizzata sia per l'addestramento che per la validazione.

Le metriche valutate sono descritte nell'apposita sezione (sez.3.3). Queste vengono misurate per ogni fold, per poi calcolarne la media.

KBANN è dotato di due parametri che permettono di gestire il comportamento del modello. Questi parametri sono chiamati **omega** e **gamma** e sono stati impostati rispettivamente a 2 e 0.LTN è stato addestrato per 50 epoche con addestramento con loss Binary Cross Entropy e 50 epoche tramite ottimizzazione con la loss del metodo LTN.

Per il modello siamese abbiamo diverse impostazioni che sono state testate durante lo svolgimento degli esperimenti. Distingueremo **"Siamese 1"** ha due parallelizzazioni, con pesi nella loss pesata impostati a 2 per i dati normali e 1 per quelli appartenenti alla conoscenza; mentre il modello **"Siamese 2"** ha tre parallelizzazioni e pesi impostati a 2.5 per gli elementi dei dati, 1 per gli elementi della conoscenza con output 0 e 2 per gli elementi della conoscenza con output 1. Per ogni modello sarà possibile scegliere di usare la generazione di dati sulla base della conoscenza, e una fase di preaddestramento sulla conoscenza. Il modello siamese verà addestrato su un massimo di 25 epoche, per evitare overfitting.

Capitolo 4

Risultati dei metodi esistenti

In questo capitolo presenteremo i risultati ottenuti durante lo svolgimento degli esperimenti, discutendone il significato e le implicazioni. Effettueremo un confronto tra i vari metodi di iniezione, e ne valuteremo il contributo nelle performance, capacità di generalizzazione e robustezza. Discuteremo, quindi, della reale utilità di questi metodi e quanto apportino un contributo positivo rispetto alle reti neurali non educate.

4.1 Metodi di iniezione esistenti

In questa sezione sono presentati i risultati di previsione di ogni metodo di SKI e del modello non educato (Uneducated), secondo le metriche descritte nella sezione 3.3. Con il solo scopo di una visualizzazione più compatta dei risultati, indicheremo con k_0 e k_1 le metriche knowledge 0 e knowledge 1. Vedremo tre tabelle, ognuna delle quali fa riferimento ad un dataset o ad una conoscenza differente. In ordine vedremo la tabella delle performance di ogni metodo su Pima (sezione 3.1.1), quella su Breast Cancer Wisconsin (sezione 3.1.2) prima con la conoscenza di DAG e successivamente quella di HN.

Model	Accuracy	Balanced	Recall	Specificity	F1	k_0	k_1
Uneducated	0.7500	**0.7463**	0.7359	0.7566	**0.6685**	1.0	0.8836
KBANN	**0.7630**	0.7153	0.5559	**0.8746**	0.6214	1.0	**1.0**
KINS	0.6888	0.7186	**0.7847**	0.6526	0.6374	1.0	0.9044
LTN	0.7148	0.7322	0.7797	0.6848	0.6534	1.0	0.9498

Tabella 4.1: Risultati dei modelli su dataset Pima

Model	Accuracy	Balanced	Recall	Specificity	F1	k_0	k_1
Uneducated	**0.9557**	**0.9535**	**0.9521**	0.9550	**0.9375**	0.9776	0.9766
KBANN	0.9485	0.9252	0.8754	**0.9749**	0.9134	**1.0000**	**1.0000**
KINS	0.9471	0.9376	0.9174	0.9577	0.9202	0.9776	0.9451
LTN	0.9313	0.9105	0.8499	0.9711	0.8887	0.9836	0.8920

Tabella 4.2: Risultati dei modelli su Breast Cancer Wisconsin DAG

Model	Accuracy	Balanced	Recall	Specificity	F1	k_0	k_1
Uneducated	**0.9557**	0.9535	0.9521	0.9550	**0.9375**	0.9776	0.9766
KBANN	0.9414	0.9276	0.8975	0.9577	0.9087	**1.0000**	**1.0000**
KINS	0.9543	**0.9565**	**0.9580**	0.9549	0.9343	0.9799	0.9748
LTN	0.9285	0.9200	0.8640	**0.9761**	0.8983	0.9849	0.8889

Tabella 4.3: Risultati dei modelli su Breast Cancer Wisconsin HN

Dalle tabelle mostrate i metodi utilizzati non sembrano fornire particolari miglioramenti alle performance del modello non educato, anzi, sembrano anche peggiorare se non in casi marginali.

KINS sembra apportare miglioramenti alla recall, che nel nostro caso di studio sarebbe preferibile, volendo minimizzare la quantità di falsi negativi. Ma in KINS questo si traduce in una riduzione della specificity, abbassando la balanced accuracy. Questo significa che il modello non sta realmente distinguendo più facilmente gli elementi della classe uno, ma sta etichettando con probabilità maggiore gli elementi con la classe uno, anche a discapito degli elementi con classe zero.

Già da questi risultati KBANN risulta il metodo che meglio rispetta la conoscenza, totalizzando un accuratezza piena su questi elementi. Ciò deriva dalla struttura stessa della rete KBANN che viene costruita interamente per rispettare la conoscenza.

4.2 Perturbazione di metodi di iniezione esistenti

In questa sezione vedremo come i modelli educati e non educati reagiscono a perturbazioni dei dati di training, per valutare se e come i metodi di SKI migliorano la robustezza dei modelli.

4.2.1 Metodi di perturbazione esistenti

In questa sezione analizziamo la variazione delle performance dei modelli rispetto a perturbazioni dei dati che in letteratura sono state già applicate su metodi di SKI [21], analizzando i pattern su dataset e conoscenze differenti.

Figura 4.1: Confronto tra reti neurali non informate e modelli informati (Uneducated e KBANN, KINS, LTN) sul dataset Pima Indians Diabetes, valutati con perturbazioni (Feature Noise, Label Flipping e Data Dropping) ad intensità crescente rispetto a metriche predittive (Accuracy, Balanced, Recall, Specificity, F1) e alla coerenza con la conoscenza (knowledge_0 e knowledge_1).

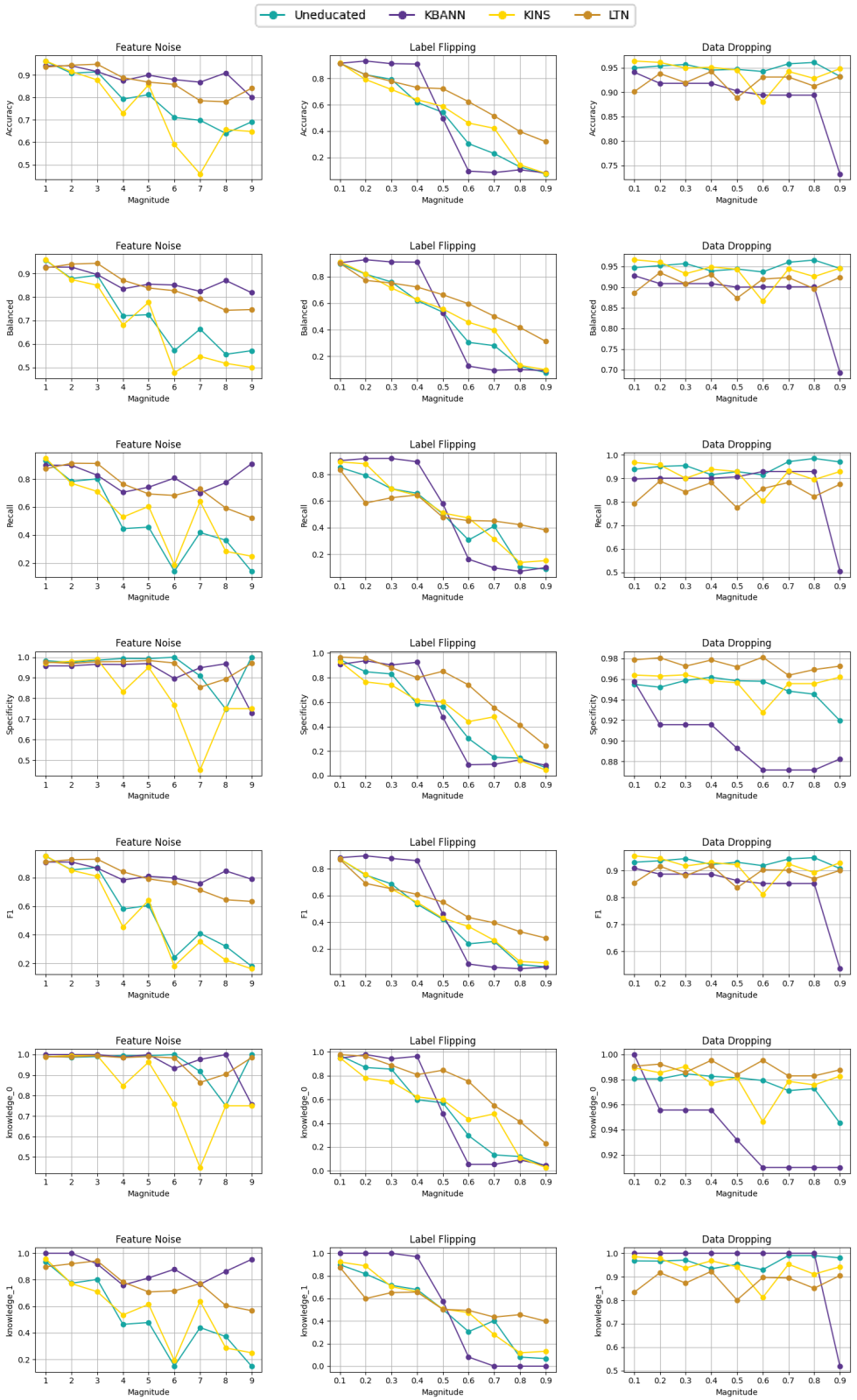


Figura 4.2: Performance dei modelli a seguito di perturbazioni su BCW con HN

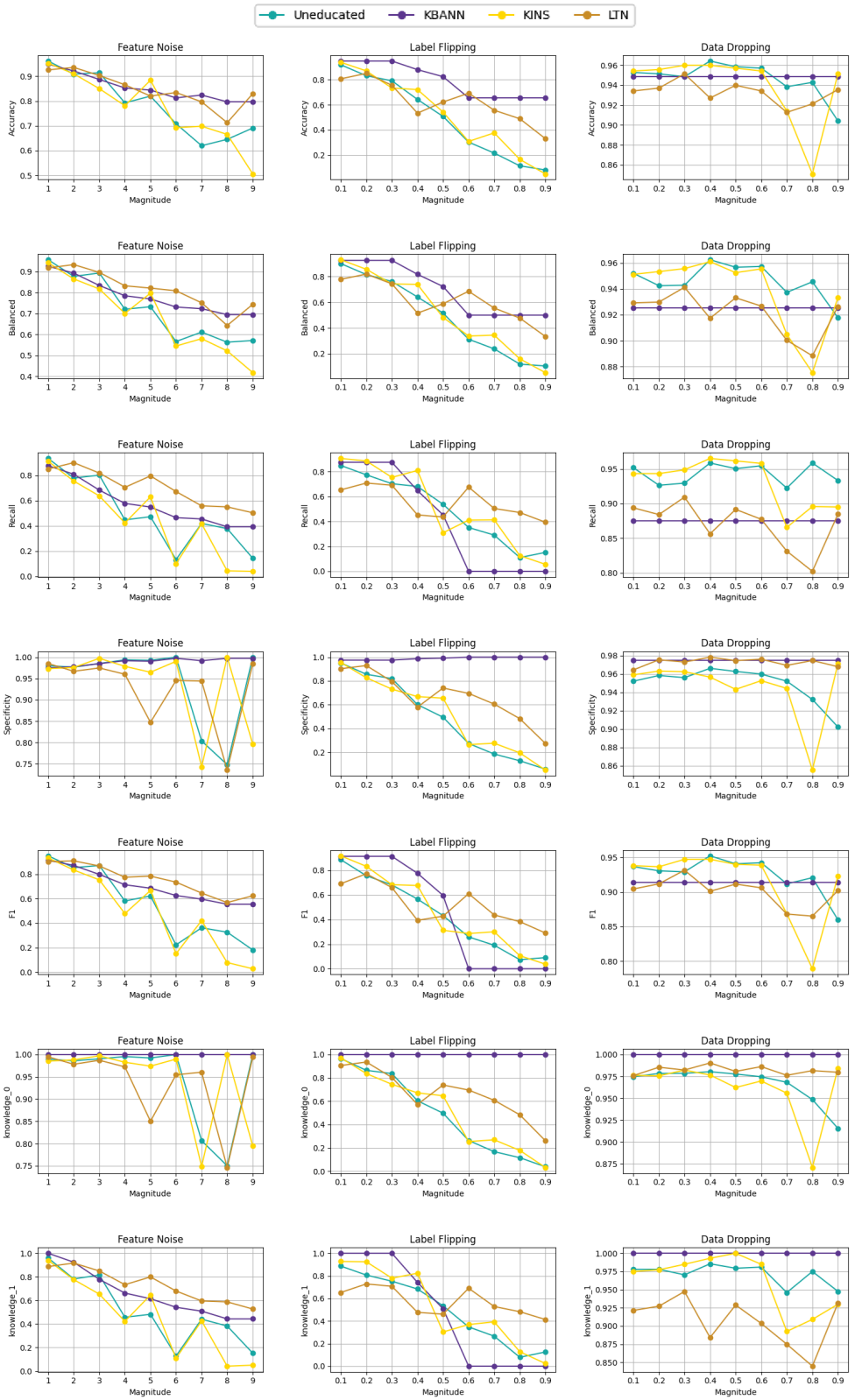


Figura 4.3: Performance dei modelli a seguito di perturbazioni su BCW con DAG

In linea generale, come potevamo aspettarci, dai grafici di entrambi i dataset vediamo che le performance di tutti i metodi tendono a calare con l'aumentare dell'entità della perturbazione. Tra i metodi indicati, quelli che apportano i miglioramenti più significativi, dal punto di vista della robustezza, sono LTN e KBANN.

Inoltre, è importante precisare che la maggior parte delle fluttuazioni che si discostano dal normale andamento decrescente delle prestazioni sono dovute a variazioni stocastiche per un numero di fold limitato o fluttuazioni improvvise dovute alla non linearità delle funzioni di attivazione utilizzate, al variare delle perturbazioni. Si è notato, infatti, che queste fluttuazioni, specialmente nelle metriche di accuratezza bilanciata (e non), tendono a scomparire all'aumentare dei fold utilizzati.

Vediamo, in figura 4.4, il risultato della balanced accuracy dei modelli in test più approfonditi e statisticamente più rilevanti, che sono stati effettuati sul dataset Pima a conferma di quanto detto prima. Questi test sono stati effettuati solo in questa occasione per confermare l'ipotesi, poiché richiedono tempi e risorse di elaborazione significativi, rendendo particolarmente restrittivi i limiti temporali del presente lavoro.

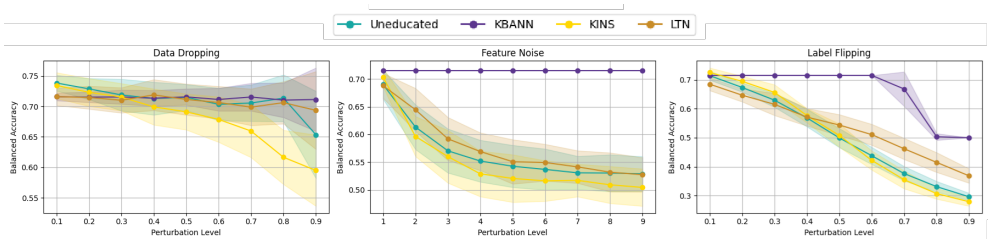


Figura 4.4: Risultato dei test di validazione condotti in crossvalidazione con 5 fold e 30 ripetizioni utilizzando 30 seed diversi per le perturbazioni, aumentando dunque la rilevanza statistica dei risultati ottenuti. Nell'immagine vediamo, per ogni metodo di iniezione e per il metodo non educato, l'andamento della balanced accuracy rispetto all'aumentare dell'entità di ogni perturbazione e la relativa deviazione standard.

Dalle immagini, vediamo che per quanto riguarda Feature Noise e Label Flipping, i risultati hanno una deviazione standard contenuta - specialmente per Label Flipping - confermando anche i risultati ottenuti precedentemente.

Per il Data Dropping, invece, vediamo che la deviazione standard aumenta all'aumentare della perturbazione. Questo potrebbe essere dovuto al fatto che, quando gli elementi eliminati sono tanti, le performance dipendono fortemente da quali elementi del dataset vengono eliminati. Eliminando elementi fondamentali per la descrizione delle classi di riferimento, risulterà più difficile per

il modello comprendere le relazioni che intercorrono tra i dati, peggiorando le performance. In altri casi, potrebbero essere eliminati elementi che aggiungono rumore al dataset, portando miglioramenti alle performance. Questo aumenta notevolmente il rumore nei risultati ottenuti con alti livelli di data dropping, aumentando la deviazione standard. Nel complesso, i risultati confermano quanto ottenuto in precedenza e inoltre dimostrano che le fluttuazioni nell'andamento dei modelli diminuiscono all'aumentare del numero dei test effettuati.

Discuteremo di seguito le implicazioni derivanti dai risultati nei grafici precedenti, che riportano l'andamento dei metodi di iniezione a seguito dell'applicazione di perturbazioni su entrambi i dataset.

Analisi del comportamento di KBANN

Solo KBANN tende ad una stazionarietà rispetto alle perturbazioni e risulterebbe da questi test il più robusto. Questa resistenza alle perturbazioni deriva, in realtà, dalla struttura stessa di KBANN che viene costruita direttamente sulla base della conoscenza. Questo lo rende particolarmente fedele a quelle che sono le regole della conoscenza, spesso a discapito delle capacità di generalizzazione.

Infatti KBANN, se pure addestrato su dati perturbati, rimanendo fedele a quelle che sono le regole della conoscenza, riesce comunque ad ottenere performance molto simili a quelle ottenute durante l'addestramento non perturbato, dato che le previsioni vengono effettuate su dati di validazione non perturbati.

Questo è anche visibile analizzando la loss di KBANN che ha un comportamento singolare se confrontato con gli altri metodi. Durante l'addestramento distinguiamo tra loss di training - ovvero quanto sbaglia il modello, in relazione al set di training, cioè quei dati su cui il modello si sta addestrando - e la loss di validazione - ovvero quanto sbaglia sul set di dati di validazione, cioè dati che il modello non ha mai visto prima.

Normalmente la loss di validazione risulta sempre maggiore o uguale alla loss di training, semplicemente perché il modello tende a fare meglio e quindi sbagliare meno sui dati che ha usato per sistemare i propri pesi durante l'addestramento.

In KBANN, la loss senza perturbazione ha tendenzialmente questo comportamento, diversamente fa quando introduciamo una perturbazione. In questo caso, infatti, vediamo una tendenza opposta, in cui la loss di validazione tende a diventare sempre più piccola rispetto alla loss di training.

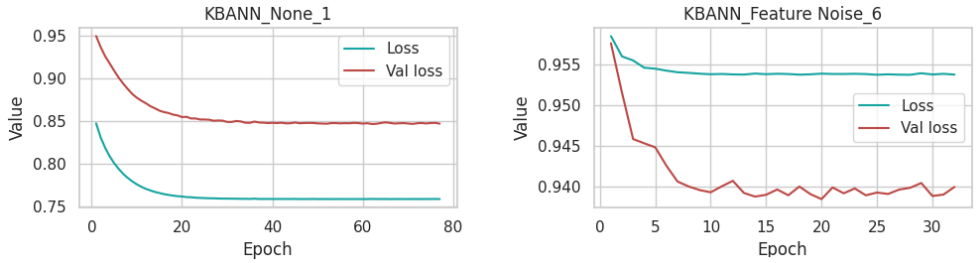


Figura 4.5: Confronto tra il decorso della loss di training e quello della loss di validazione di KBANN. Il grafico *KBANN_None_1*, indica il caso in cui nessuna perturbazione viene applicata; mentre, il grafico *KBANN_Feature Noise_6* mostra il test il cui viene applicata la perturbazione Feature Noise con intensità impostata al livello 6. L'immagine vuole mostrare come la loss di validazione diventi minore di quella di training a seguito della perturbazione.

Nei grafici in figura 4.5, vediamo l'andamento di entrambe le loss nel tempo durante l'addestramento. A sinistra (KBANN_None_1) abbiamo l'andamento delle loss senza nessuna perturbazione, mentre a destra (KBANN_Feature_Noise_6) abbiamo l'andamento delle loss con perturbazione Feature Noise ed entità 6.

Risalta immediatamente che le loss si scambiano di posizione dopo l'aggiunta della perturbazione, quindi la loss di validazione diventa più piccola della loss di training. Aumentando la perturbazione si è notato che questo fenomeno tende a diventare sempre più marcato, in quanto la loss di training tende ad aumentare sempre di più, staccandosi da quella di validazione. Questo ci fa capire che durante l'addestramento KBANN risente della perturbazione, ma durante la validazione (dove i dati non sono perturbati) riesce comunque ad effettuare delle previsioni corrette basandosi sulla conoscenza.

A conferma dell'ipotesi, ulteriori test sono stati effettuati su KBANN, perturbando anche i dati di validazione. Quello che ci aspettiamo, è che le performance calino, specialmente quelle sulla conoscenza. Dovrebbero calare le performance di almeno una delle due classi. Infatti, è proprio quello che succede, come visibile in figura 4.6.

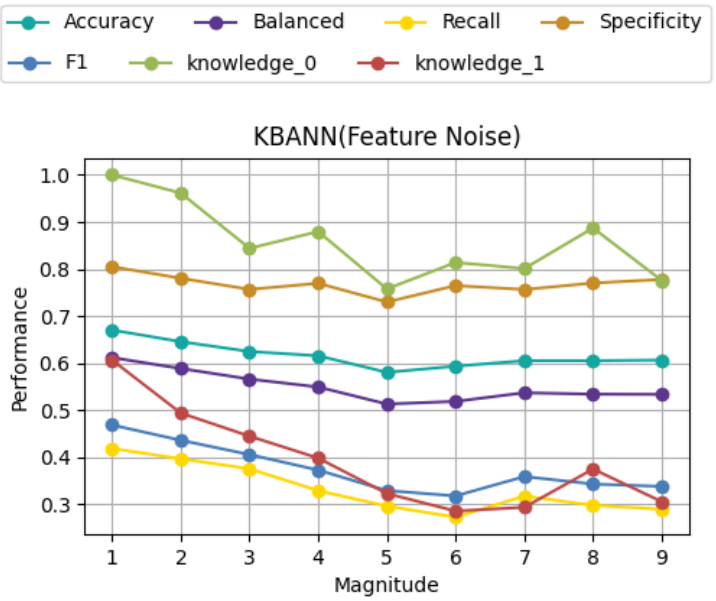


Figura 4.6: Grafico che mostra il decorso delle performance di KBANN a seguito di un livello crescente di Frature Noise, perturbando allo stesso modo sia il set di training che quello di validazione.

Tra i metodi di perturbazione, è stata visualizzata solo la Feature Noise, poichè è l'unica concettualmente valida da considerare per la perturbazione dei dati di validazione. Risulta appropriato, infatti, pensare che anche i dati di validazione siano rumorosi in un'applicazione reale. Ma allo stesso tempo considerare dati di validazione con etichette invertite significherebbe, compromettere la funzione stessa del validation set, che è quella di fornire una stima attendibile delle prestazioni del modello su dati corretti ma mai visti. Introducendo etichette errate, si perderebbe la possibilità di valutare correttamente la capacità del modello di generalizzare nonostante sia addestrato su dati errati, rendendo il processo di validazione non informativo o fuorviante. Per il data dropping, invece, la perturbazione dei dati di validazione, non comporterebbe una variazione delle perforamnce che possa essere attribuibile alla perturbazione stessa, in quanto si tratterebbe semplicemente di effettuare la previsione su meno elementi.

I risultati ottenuti confermano l'ipotesi, dato che vediamo le performance di KBANN calare all'aumentare della perturbazione. In particolare vediamo che sia la recall, che le performance su elementi della conoscenza di classe uno, calano drasticamente, facendo calare di conseguenza anche la balanced accuracy, che tende al 50%. KBANN, quindi, resta molto vincolato alla conoscenza, garantendo ottime performance in caso in cui ad essere perturbati siano i dati

di training, ma a discapito della generalizzazione. Questo vincolo con la conoscenza, può essere reso più o meno influente in base ai parametri omega e gamma, ma le capacità di generalizzazione di KBANN potrebbero comunque risultare limitate a causa dell'eccessiva semplicità della rete stessa.

KBANN risulta comunque un metodo valido, che riesce con dataset semplici e ben descritti dalla conoscenza ad ottenere buone performance con un'ottima robustezza.

Tendenze ambigue delle performance

Importante nell'analisi di questi risultati, è considerare le performance nel complesso e non soffermarsi sulle singole metriche. Infatti alcune di queste sembrerebbero addirittura aumentare, con entità di perturbazione maggiore. Prendendo come esempio l'accuracy e la specificity di **KINS**, nel dataset **Pima**, in figura 4.1, nel caso della **Feature Noise**, queste sembrano addirittura aumentare. Questo fenomeno è prettamente dovuto allo sbilanciamento del dataset, che ha più elementi con target zero. All'aumentare della perturbazione, il modello tende a restituire un numero crescente di zeri in output. Questo comportamento incrementa la specificity, ovvero la capacità del modello di riconoscere correttamente gli elementi della classe zero. Tuttavia, a questo aumento corrisponde una diminuzione della recall, cioè la capacità di identificare correttamente gli elementi della classe uno. Il fenomeno raggiunge un punto critico in cui la specificity è massima e la recall minima. Di conseguenza, anche l'accuracy complessiva aumenta, ma questo effetto è in parte fuorviante: poiché il dataset è sbilanciato, prevedere più zeri porta a un'apparente miglioramento della prestazione complessiva per una questione meramente probabilistica. Questo squilibrio è evidenziato dalla balanced accuracy, che invece mostra un calo progressivo, indicando una perdita di equilibrio nel riconoscimento tra le due classi.

Lo stesso si verifica nel caso del Data Dropping in cui la recall sembra aumentare all'aumentare della perturbazione. Questo può, in qualche modo, ricondursi sempre allo sbilanciamento del dataset. Infatti, rimuovendo elementi in modo puramente casuale, avremo molta più possibilità di rimuovere elementi con classe zero (essendo questi più numerosi) portando ad un naturale bilanciamento delle classi. Si può vedere, infatti, che al crescere della recall, anche in questo caso la specificity decresce.

Confronto tra le perturbazioni

La perturbazione che sembra compromettere maggiormente le performance è il Label Flipping, in quanto si osserva una balanced accuracy che raggiunge i valori più bassi. Questo fenomeno, in realtà, indica semplicemente che i modelli

tendono a classificare gli elementi come appartenenti alla classe opposta, mantenendo un livello di accuratezza che potrebbe essere quasi specchiato intorno il livello 0.5 di Label Flipping. Ad esempio, si registrano performance intorno al 10% laddove inizialmente erano del 90%, e analogamente performance del 30% dove in precedenza erano intorno al 70%. Se dunque si invertissero le etichette a posteriori, si otterrebbero risultati molto simili a quelli originari.

Nel caso di KBANN, come illustrato in figura 4.3, si osserva che, oltre la soglia di 0.5 nella perturbazione, che è apparentemente più vantaggioso. In realtà, il modello restituisce sistematicamente la classe zero, ottentendo specificity pari a 1 e recall pari a 0, la cui media è 0.5. Questo, evidenzia esclusivamente l'incapacità di distinguere correttamente tra le due categorie.

Paradossalmente, potrebbe risultare più vantaggioso ottenere una balanced accuracy inferiore al 50%, anche se, a livello grafico potrebbe apparire migliore il metodo che mantiene tale valore costante. Infatti, una balanced accuracy pari a 0 implica che il modello sta classificando sistematicamente tutti gli elementi nella classe sbagliata. Tuttavia, questa situazione suggerisce che il modello ha appreso una logica coerente, seppur invertita: basterebbe infatti invertire le sue previsioni per ottenere una balanced accuracy del 100%.

I metodi che riescono meglio a mantenere una certa capacità di distinzione tra le classi, anche a livelli relativamente elevati di Label Flipping, sono KBANN ed LTN che risultano quelli più restii ad invertire le etichette, ottenendo le perfomance più alte per livelli di Label Flipping minori a 0.5 (o anche più). Allo stesso tempo, però, per livelli maggiori, potrebbe essere corretto constatare che, come spiegato in precedenza, gli altri modelli presentano delle performance preferibili anche se teoricamente peggiori. Questa è, in applicazioni pratiche, una situazione più unica che rara, e considereremo come generalmente preferibili le performance di KBANN ed LTN.

Non considerando il Label Flipping, la perturbazione che peggiora maggiormente le performance è Feature Noise. In tutti e tre i casi, infatti, il modello non educato e KINS arrivano a performance prossime al 50% per la balanced accuracy, indicando che il modello non riesce, come detto prima, a distinguere gli elementi delle due classi. Anche in questi casi, i modelli che performano meglio sono KBANN ed LTN, confermando la loro robustezza.

Il Data Dropping, invece, non sembra perturbare particolarmente le performance, rimanendo quasi invariate fino a un drop rate molto elevato. Questo potrebbe essere attribuito a dati particolarmente semplici, in cui le relazioni che i modelli riescono a trovare possono essere rappresentate correttamente anche con pochi elementi.

Questa è, però, l'unica perturbazione in cui LTN sembra performare peggio degli altri metodi e sembra essere una tendenza generale, in quanto si ripete per tutti i dataset e conoscenze.

Confronto tra metriche

Le metriche che maggiornmente sembrano risentire delle perturbazioni sono quelle legate agli elementi della classe uno. Quindi sia la recall che la knowledge_1 sembrano peggiorare di più all'aumentare delle perturbazioni. Questo può sempre essere collegato con lo sbilanciamento del dataset in quanto, in caso di difficoltà nel generalizzare, potrebbe essere preferibile per il modello restituire la classe con numerosità maggiore.

L'unica performance per cui questa tendenza sembra essere opposta è nel caso del Data Dropping, in cui specialmente nel dataset Pima, come visto in precedenza, vediamo un importante incremento della recall all'aumentare della perturbazione. Questo, però, si traduce sempre in un calo della specificity.

4.2.2 Nuovi metodi di perturbazione

Oltre a metodi di perturbazione esistenti, è stato testato un nuovo metodo di perturbazione chiamato Feature Corruption. Questo è basato su due parametri di perturbazione: l'entità intesa come probabilità di corruzione e l'entità intesa come l'intensità della corruzione. Visualizzeremo, quindi, dei grafici che permettano di visualizzare come le performance cambiano in relazione a questi due parametri.

Per compattezza mostreremo solo dei grafici in cui si mettono a confronto i vari metodi di iniezione rispetto alla balanced accuracy.

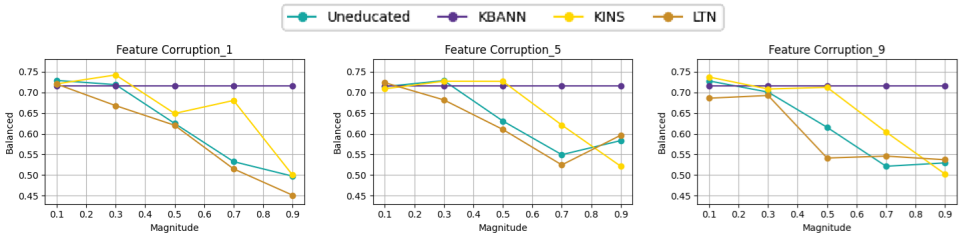


Figura 4.7: Grafici che mostrano il decorso della Balanced Accuracy di modelli educati e non educati (Uneducated e KBANN, KINS, LTN), rispetto a livelli crescenti di Feaure Corruption. Ogni grafico si chiamerà "Feature Corruption_N" dove 'N' indica l'intensità della corruzione e, quindi, quanto ci allontaniamo dal range di riferimento di ogni feature, mentre sull'asse delle x la probabilità di corruzione.

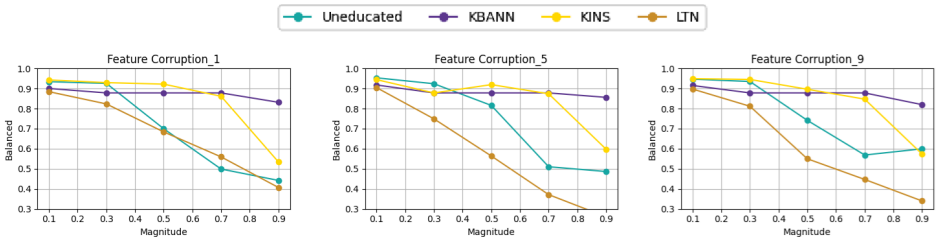


Figura 4.8: Feature Corruption di Breast Cancer con HN

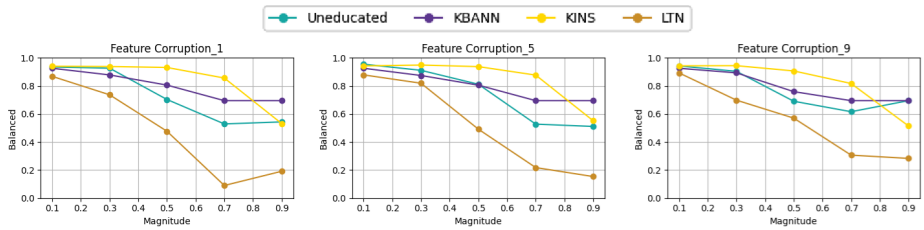


Figura 4.9: Feature corruption di Breast Cancer con DAG

Dai risultati si evince che la distanza della corruzione incide poco sui risultati. Questo potrebbe derivare da come a seguito della perturbazione i punti vengono distribuiti. Infatti, quello che feature corruption fa, è spostare dei punti rispetto ad una o più feature, impostandole ad un determinato valore che sia a destra o a sinistra rispetto al range scelto (vedi sez.3.6). Questo causa dei punti distribuiti più o meno casualmente, lontano dal normale spazio in cui i punti sono distribuiti. Questo non darà troppa possibilità alla rete di apprendere delle relazioni per questi punti, indipendentemente da quanto siano distanti dal normale range. Inoltre, avendo output sigmoidale, la derivata per valori estremi tende a zero, facendo si che l'output cambi poco, indipendentemente da quanto ci allontaniamo.

Maggiore importanza assume, invece, la probabilità di corruzione, che degrada concretamente le performance. Tra i modelli proposti, in questo caso, (tralasciando KBANN per i motivi discussi in precedenza) spicca KINS che, se pur non migliorando le performance, riesce ad apportare un discreto miglioramento della robustezza rispetto al modello non educato, e dimostra in tutti i test di riuscire a resistere meglio alla corruzione delle feature.

Capitolo 5

Risultati dei nuovi metodi di iniezione

In questa sezione analizzeremo le performance dei nuovi metodi di iniezione proposti e sperimentati nel corso di questo studio.

I risultati ottenuti sono stati complessivamente molto promettenti. Verranno presentati alcuni degli esperimenti condotti, evidenziando come le varie modifiche introdotte abbiano consentito di passare da performance inizialmente limitate a risultati finali sostanzialmente migliorati, dimostrando il potenziale dei metodi sviluppati.

5.1 Performance

Inizieremo con i test effettuati con rete siamese, senza generazione di dati aggiuntivi. Da questa base analizzeremo le performance modificando il numero di parallelizzazioni. Per compattezza mostreremo, per i risultati intermedi, solo i grafici relativi al dataset Pima, mostrando i miglioramenti ottenuti. Tutti i risultati del nuovo metodo mostrati di seguito, sono stati ottenuti con 20-25 epoche, poiché si è osservato che dopo questo numero di epoche il modello tende ad andare in overfitting.

Model	Accuracy	Balanced	Recall	Specificity	F1	k_0	k_1
Uneducated	0.7500	0.7463	0.7359	0.7566	0.6685	1.0000	0.8836
KBANN	**0.7630**	0.7153	0.5559	**0.8746**	0.6214	1.0000	**1.0000**
KINS	0.6888	0.7186	0.7847	0.6526	0.6374	1.0000	0.9044
LTN	0.7148	0.7322	0.7797	0.6848	0.6534	1.0	0.9498
Siamese 1	0.7617	**0.7666**	0.7994	0.7339	**0.6990**	1.0	0.9496
Siamese 2	0.7382	0.7512	**0.8004**	0.7021	0.6788	1.0	0.9576

Tabella 5.1: Modello siamese 1 e 2 senza augmentation su dataset Pima

Nella tabella 5.1 vediamo i migliori risultati ottenuti dal modello siamese impostato come descritto nella sezione 3.8.

In entrambi i casi vediamo che il metodo STI ha apportato un discreto miglioramento delle performance anche quando paragonato con il metodo non educato, specialmente nel caso di Siamese 1, che riesce ad ottenere la migliore balanced accuracy. Mentre, Siamese 2 riesce ad ottenere la migliore recall, con una balanced accuracy leggermente maggiore rispetto a quella del modello uneducated e di tutti gli altri metodi di iniezione.

5.2 Robustezza

Per quanto riguarda la robustezza del modello, vedremo dei grafici come i precedenti che mettono i vari modelli a confronto. Per compattezza, in questo caso, visualizzeremo solo il confronto della balanced accuracy.

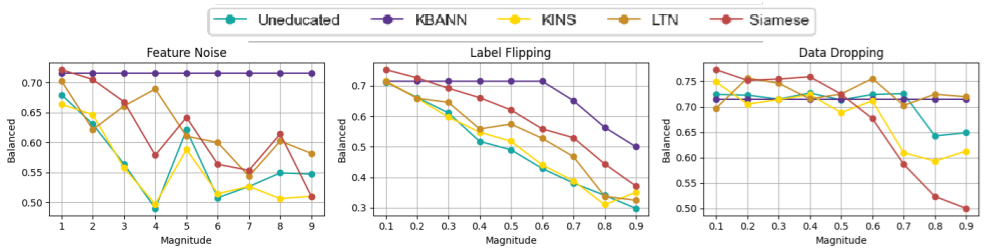


Figura 5.1: Siamese 1 con perturbazioni su pima

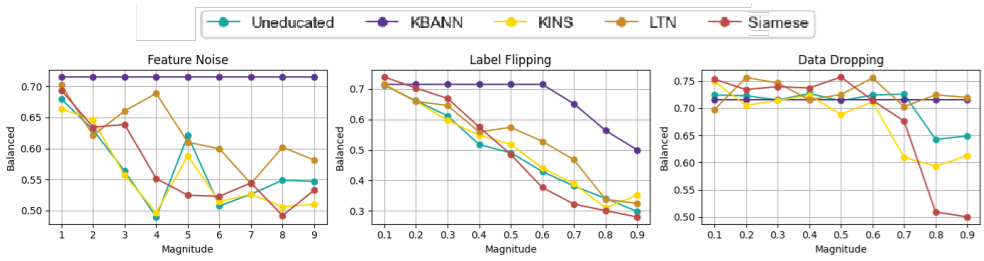


Figura 5.2: Siamese 2 con perturbazioni su pima

Dalla figura 5.1 vediamo che il modello riesce ad apportare un contributo anche dal punto di vista della robustezza relativamente alle perturbazioni. Ad eccezione di KBANN, per i motivi sopra specificati, STI riesce ad apportare già un contributo superiore agli altri metodi almeno in parte dei risultati. Nella Feature Noise LTN si alterna durante tutto il percorso con STI, ottenendo specialmente nella parte finale performance molto simili.

Per il Label Flipping (sempre escludendo KBANN) il migliore risulta STI che tiene in generale delle performance più elevate, e inverte le etichette solo dal livello di perturbazione di 0.8.

Nel caso della figura 5.2 vediamo che il contributo di Siamese 2 è notevolmente minore dal punto di vista della robustezza, portando un leggero miglioramento nella Feature Noise e nel Lable Flipping, rispetto al modello non educato.

Per quanto riguarda data dropping, invece, generalmente le performance di STI tendono a risentirne molto di più rispetto a tutti gli altri modelli, tranne nella fase iniziale in cui riesce comunque a garantire performance migliori. Questo deriva dal fatto che, utilizzando i soli dati a disposizione per descrivere la conoscenza, nel momento in cui ne riduciamo il numero, questa non viene descritta appropriatamente, dando troppa importanza ai pochi elementi della conoscenza presenti (che vengono replicati per combaciare con la lunghezza del set di training), causando un repentino calo delle performance. In alcuni test con data dropping, poteva anche succedere che il modello non riuscisse nemmeno a raggiungere fine corsa, in quanto non vi erano elementi della conoscenza sufficienti per descrivere l'una o l'altra classe. Questo risultato è confermato anche dai test su breast cancer, in cui la conoscenza identifica un numero di elementi significativamente maggiore, e le performance calano solo intorno al 90% di data dropping.

5.3 Integrazione dell'Augmentation

Per contrastare queste problematiche faremo affidamento al data augmentation tramite generazione di elementi della conoscenza, come descritto nella sezione 3.5.1.

Model	Accuracy	Balanced	Recall	Specificity	F1	k_0	k_1
Uneducated	0.7500	0.7463	0.7359	0.7566	0.6685	1.0000	0.8836
KBANN	**0.7630**	0.7153	0.5559	**0.8746**	0.6214	1.0000	**1.0000**
KINS	0.6888	0.7186	0.7847	0.6526	0.6374	1.0000	0.9044
LTN	0.7148	0.7322	0.7797	0.6848	0.6534	1.0	0.9498
Siamese 1	0.7382	**0.7521**	**0.8001**	0.7042	**0.6767**	1.0	0.9869

Tabella 5.2: Modello siamese 1 con augmentation su dataset Pima

In questo caso, vediamo che il metodo siamese con l'aggiunta di dati generati tramite la conoscenza, raggiunge la miglior recall e balanced accuracy. Riesce, inoltre, ad apportare un importante miglioramento anche dal punto di vista della robustezza.

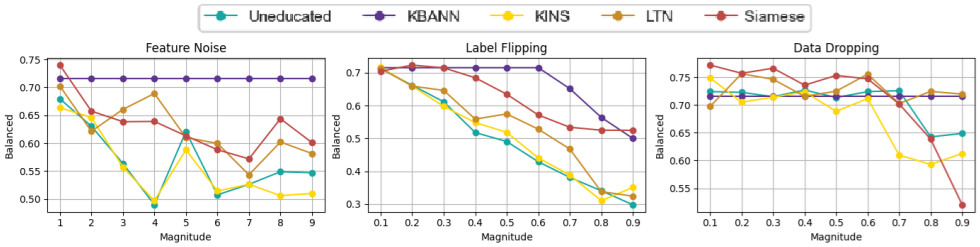


Figura 5.3: Modello siamese 1 con augmentation su dataset Pima perturbato

In questo caso, migliora la robustezza di STI anche se confrontato con LTN. C'è solo una fluttuazione di LTN nella fase iniziale del Feature Noise, in cui performa meglio di STI. Del resto, STI migliora molto le sue performance, specialmente nel Label Flipping in cui nella fase iniziale si comporta quasi come KBANN, mentre nella parte finale non inverte mai le proprie etichette, riuscendo comunque a rimanere al di sopra del 50% di balanced accuracy. Risulta migliorato anche nel Data Dropping, in cui le performance iniziano a calare dall'entità 0.6 (rispetto al 0.4 del test precedente). Però anche in questo caso risulta sensibile ad alti livelli di Data Dropping.

5.4 Modifica della pipeline di addestramento

Per risolvere questo problema, è stato introdotto un'ulteriore miglioramento. La pipeline di addestramento è stata divisa in due parti: una in cui il modello viene addestrato in modo classico ma esclusivamente sugli elementi sintetici della conoscenza, portando in sole 10 epoche ad ottenere il 100% dell'accuratezza sulla conoscenza; e la seconda in cui avviene l'addestramento siamese. L'idea è che questo metodo ci permette di trovare un minimo nella funzione di costo per la conoscenza, per poi muoverci, successivamente, nell'intorno di questo minimo (rimanendo vincolati in tutto il processo alla conoscenza stessa). Questo ci permette di trovare le migliori performance generali, tenendo alte quelle della conoscenza.

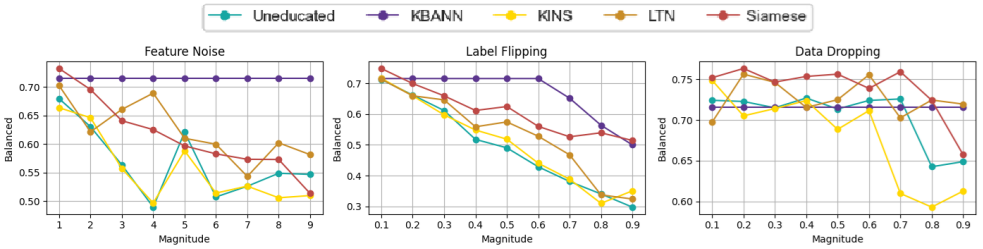


Figura 5.4: Modello siamese 1 con augmentation e preaddestramento su dataset Pima perturbato

Model	Accuracy	Balanced	Recall	Specificity	F1	k_0	k_1
Siamese 1	0.7239	0.7485	0.8482	0.6488	0.6801	1.0	0.9687

Tabella 5.3: Modello siamese 1 con augmentation e preaddestramento su dataset Pima

In questo caso le performance restano pressoché invariate, avendo aumentato la recall ma diminuito la balanced accuracy. Però, appoggiandosi più fermamente sulla conoscenza, si è ottenuto un incremento della robustezza, a tal punto che risulta il più robusto anche nei casi in cui prima risultava più sensibile, come nel Data Dropping.

5.5 Restringimento range di generazione

Modificando il metodo di generazione della conoscenza, riusciamo a migliorare ulteriormente la robustezza nel caso della feature noise, in quanto il rumore a cui è soggetta la generazione dei dati viene attenuata.

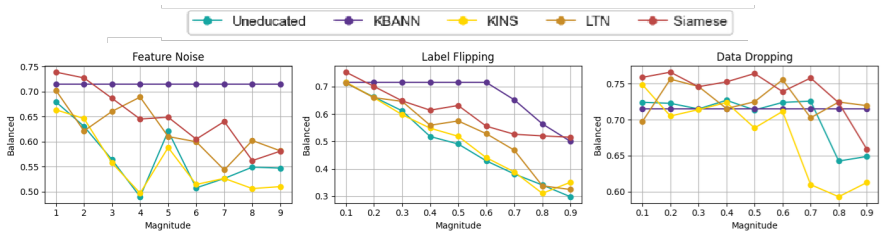


Figura 5.5: Modello siamese 1 con augmentation ridotta e preaddestramento su dataset Pima

Vediamo che tra le perturbazioni quella che ha ottenuto maggiore beneficio da questa modifica è proprio la Feature Noise. Questo effetto è visibile in modo ancora più marcato sul dataset breast cancer wisconsin.

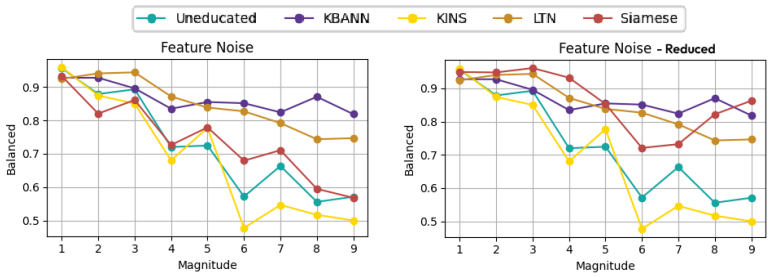


Figura 5.6: Confronto delle prestazioni rispetto alla feature noise senza riduzione (a sinistra) e con riduzione (a destra), con augmentation e un preaddestramento di 5 epoche.

5.6 Rimozione elementi contrari alla conoscenza

Come ultimo miglioramento, considerando che STI è in base ai dati che riesce ad adattarsi alla conoscenza, andremo ad eliminare tutti quegli elementi che non la rispettano. Elimineremo, quindi, tutti quegli elementi le cui feature rientrano nei vincoli della conoscenza, ma hanno target opposto. Questo approccio può risultare particolarmente utile quando la conoscenza è verificata e certa. In questo modo, riusciamo a correggere tutti quegli errori che possono avvenire nelle fasi precedenti all'addestramento.

5.6.1 Risultati su Pima

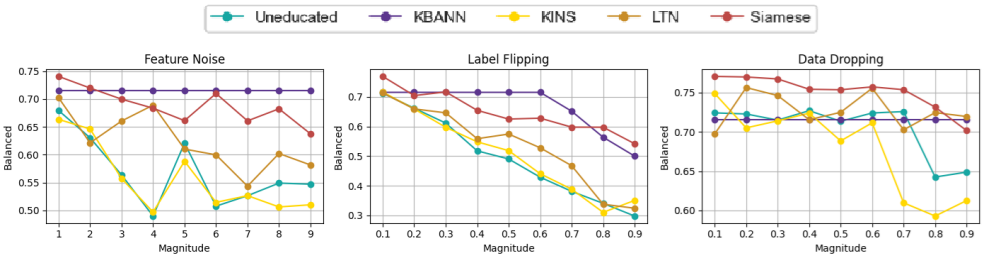


Figura 5.7: Modello siamese 1 con augmentation ridotta, preaddestramento e pulizzia dei dati su dataset Pima

Dall'immagine 5.7, vediamo che la robustezza aumenta notevolmente, risentendo sempre meno delle perturbazioni. Nel caso del Data Dropping, STI risulta avere le performance migliori tra i metodi, fino ad un drop rate di 0.8, con performance, molto simili a quelle senza perturbazione. Le performance di STI si avvicinano sempre di più a quelle di KBANN pur basandosi su una rete generica, garantendo robustezza e capacità di generalizzazione. Anche le performance sulla conoscenza senza perturbazioni, infatti, a questo punto sono simili a KBANN, ma mantenendo allo stesso tempo una recall e balanced accuracy maggiori. Inoltre, come vediamo in tabella 5.4, è l'unico metodo, a parte KBANN che raggiunge il massimo delle performance sugli elementi della conoscenza.

Model	Accuracy	Balanced	Recall	Specificity	F1	k_0	k_1
Uneducated	0.7500	0.7463	0.7359	0.7566	0.6685	1.0	0.8836
KBANN	**0.7630**	0.7153	0.5559	**0.8746**	0.6214	1.0	1.0
KINS	0.6888	0.7186	0.7847	0.6526	0.6374	1.0	0.9044
LTN	0.7148	0.7322	0.7797	0.6848	0.6534	1.0	0.9498
Siamese 1	0.7460	**0.7629**	**0.8244**	0.7014	**0.6935**	1.0	1.0

Tabella 5.4: Modello siamese 1 con augmentation su dataset Pima

5.6.2 Risultati su Breast Cancer Wisconsin

Un risultato interessante si verifica con il dataset breast cancer wisconsin in cui il modello iniettato con STI sembra addirittura non risentire della perturbazione Lable Flipping, risultato che si verifica per entrambe le conoscenze.

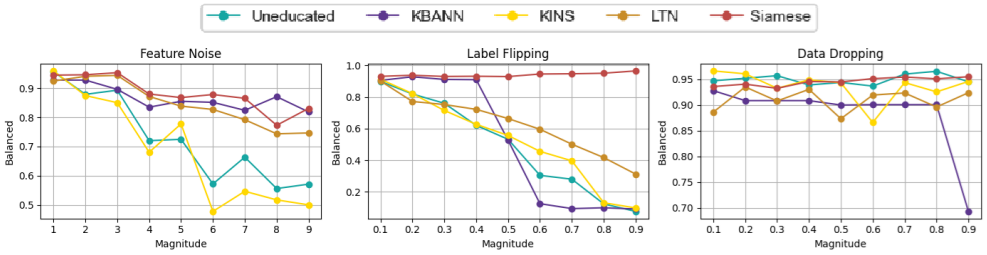


Figura 5.8: Modello siamese 1 con augmentation ridotta, preaddestramento e pulizzia dei dati su dataset Breast Cancer con conoscenza HN

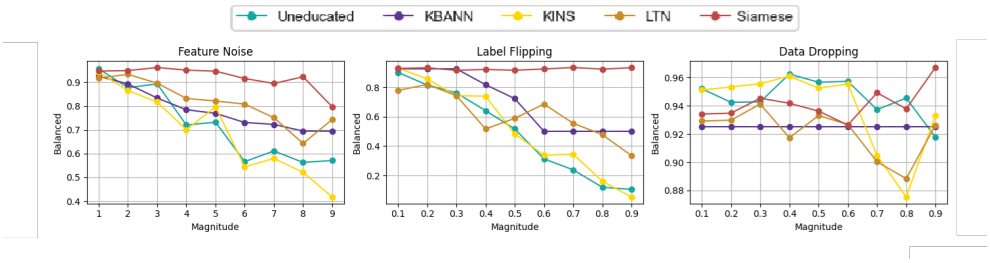


Figura 5.9: Modello siamese 1 con augmentation ridotta, preaddestramento e pulizzia dei dati su dataset Breast Cancer con conoscenza DAG

Dalle immagini, è visibile l'effetto descritto precedentemente. Questo è dovuto principalmente alla pulizia dei dati, dove rimuovendo tutti quegli elementi della conoscenza che non rispettano le regole, riusciamo comunque a tenere alta la coerenza con la conoscenza. Nel caso specifico del dataset breast cancer, la conoscenza ha performance molto elevate nella prevevisione degli elementi. Con livelli di Label Flipping elevati, infatti, le performance tenderanno a quelle della conoscenza. Questo spiega anche perché in Pima non abbiamo lo stesso risultato, dato che la conoscenza ha, in questo caso, un'accuratezza minore.

Inoltre, in entrambi i casi la balanced accuracy aumenta all'aumentare della perturbazione Data Dropping. Questo potrebbe essere dovuto, come spiegato in precedenza, alla rimozione randomica che avviene nel processo di Data Dropping che rimuove statisticamente più elementi della classe zero che quelli appartenenti alla classe uno. Questo potrebbe essere confermato dal fatto che a crescere tra la specificity e la recall, è proprio la seconda in entrambi i casi.

5.7 Risultati STI con Feature Corruption

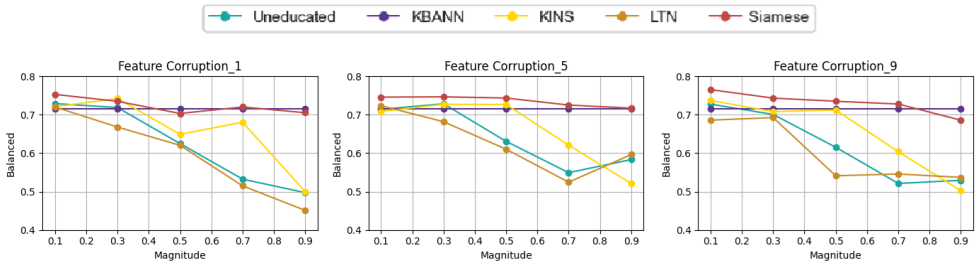


Figura 5.10: Siamese 1 con Feature Corruption su Pima

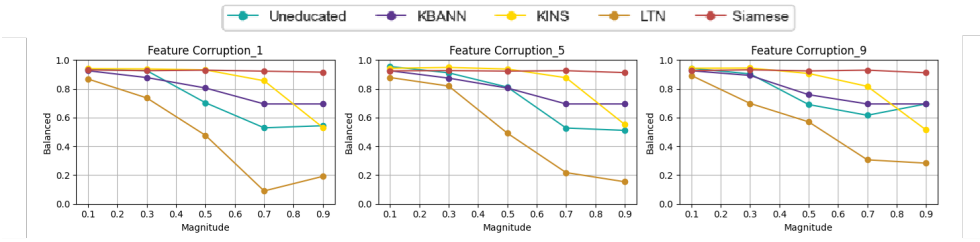


Figura 5.11: Siamese 1 con Feature Corruption su Breast Cancer con conoscenza DAG

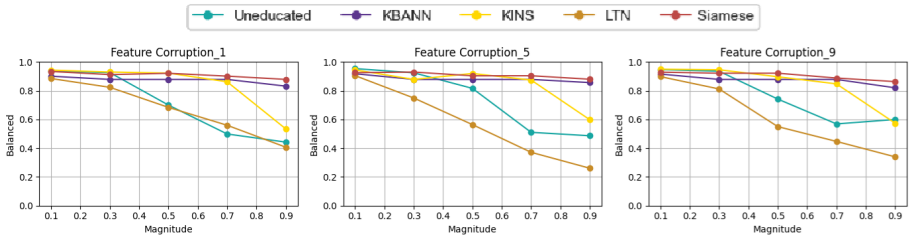


Figura 5.12: Siamese 1 con Feature Corruption su Breast Cancer con conoscenza HN

In figura 5.10 vediamo come Siamese 1 si comporta a seguito della perturbazione Feature corruption, ottenendo prestazioni simili a KBANN, risultando anche in questo caso tra i più robusti e migliorando le performance del modello non educato.

Nel caso di Breast Cancer, invece, performa meglio anche di KBANN e non sembra risentire particolarmente della perturbazione. Le performance subiscono un calo quasi impercettibile e tra le conoscenze in questo caso, DAG è quella

che fornisce resistenza maggiore a STI.

Questa resistenza, anche per livelli elevati di Feature Corruption, può anche essere attribuita alla rimozione degli elementi inconsistenti, poiché verranno rimossi tutti quegli elementi perturbati che risultano contrari alla conoscenza.

5.8 Altri risultati

Un altro risultato interessante è che STI sembrerebbe ridurre le epoche necessarie per convergere. Nel modello non educato la loss di validazione sembra convergere tra le 40/50 epoche, mentre Siamese 1 converge tra le 20/25 epoche, migliorando le prestazioni di addestramento.

I risultati riportati risultano sufficienti per dimostrare le potenzialità del metodo proposto, ma risultati più dettagliati e altri non presenti in questa sezione sono presenti nel repository personale del candidato:
https://github.com/NunzioDA/TirocinioMagistrale.

Capitolo 6

Conclusioni

Nel corso del presente lavoro abbiamo analizzato le capacità di diversi metodi di SKI presenti in lettereatura, studiandone pregi e difetti. Abbiamo, valutato sia il contributo dal punto di vista delle performance, valutando diverse metriche per dataset sbilanciati; ma abbiamo anche valutato il contributo in termini di robustezza che questi metodi riescono a dare, valutandone le performance a seguito di perturbazioni crescenti.

I metodi di SKI sono ancora poco utilizzati e poco esplorati in ambito clinico, ma potrebbero essere la chiave per la risoluzione di diversi problemi relativi all'applicazione di tecnologie di AI in ambito medico.

Inoltre, studi precedenti su metodi di iniezione hanno esplorato poco il contributo in termini di robustezza che questi metodi possono apportare, nonostante sia un contributo fondamentale in contesti clinici in cui i dati possono essere rumorosi e scarsi.

I metodi esistenti hanno apportato miglioramenti limitati alle performance e alla robustezza del modello di base. Tra i metodi di SKI presentati, quelli che si sono distinti maggiormente, apportando un contributo positivo nelle performance o alla robustezza sono LTN per le perturbazioni esistenti e KINS per la nuova perturbazione presentata.

KBANN è un metodo che si è rivelato particolarmente robusto e resistente alle perturbazioni. Ciò deriva da come KBANN crea la propria rete sulla base della conoscenza e da come la rispetta più o meno fermamente in base ai parametri omega e gamma. Questo suo punto di forza è, in realtà, anche la sua debolezza, in quanto la dimensione e complessità della rete è relazionata esclusivamente alle regole della conoscenza. Sarà, quindi una rete sempre troppo piccola per generalizzare su dati più complessi e farà difficoltà a trovare quelle che sono le relazioni tra i dati più articolate. Di conseguenza KBANN può

rivelarsi un ottimo strumento se applicato su dati semplici e in contesti rumorosi, in cui può riuscire ad apportare un contributo favorevole resistendo ad eventuali perturbazioni, mentre su dataset più complessi potrebbe non essere la scelta più appropriata.

Confrontando le due conoscenze utilizzate sul dataset Breast Cancer Wisconsin, hanno apportato poca differenza ai risultati complessivi, sia dal punto di vista delle performance che dal punto di vista della robustezza dei modelli confrontati con il modello non educato, il che supporta la generalità dei risultati ottenuti.

In questo studio è stato inoltre proposto e testato un nuovo metodo di iniezione di conoscenza che si basa sull'applicazione dell'addestramento siamese, addestrando contemporaneamente il modello sui dati classici e dati che descrivono la conoscenza. Sono state testate diverse configurazioni del metodo, partendo dal metodo più semplice dove la conoscenza è descritta estrapolando dati dal dataset fornito, fino ad integrare data augmentation, preaddestramento su elementi della conoscenza e la pulizia del dataset rispetto alla conoscenza stessa.

Il nuovo metodo ha apportato diversi miglioramenti, riuscendo ad aumentare le performance e la robustezza del modello non educato. Il metodo è riuscito, inoltre, ad apportare miglioramenti rispetto al numero di epoche necessarie per convergere. Questo risultato si è notato anche dove le performance risultavano simili a quelle del modello non educato. In generale, l'impatto del modello è positivo e non presenta limiti di generalizzazione, in quanto si basa su una rete generica di cui non modifica la struttura.

Tuttavia il metodo deve essere ulteriormente esplorato, e la sua validità deve essere ancora messa alla prova. Studi futuri potrebbero testare il metodo con modelli e dati più complessi per comprendere le concrete potenzialità in reali applicazioni. Si deve, inoltre, trovare un metodo sistematico e strutturato per impostare i pesi che danno più o meno importanza ai dati rispetto alla conoscenza, a cui il metodo è particolarmente sensibile. Inoltre, si potrebbero proporre nuovi metodi per generare elementi appartenenti alla conoscenza o per integrare dati sintetici con i dati grezzi. Si può pensare ad altri modi per dare peso diverso ad ogni regola della conoscenza, come per esempio generare un numero diverso di elementi per ogni regola o output. Inoltre, poiché il metodo è una tecnica di addestramento applicabile a qualsiasi rete, potremmo pensare di applicare il metodo siamese agli altri metodi di SKI, cercando di unire i vantaggi e ottenere performance migliori. Altri test potrebbero concentrarsi su nuove funzioni di costo, che puntano a migliorare ulteriormente le proprietà del modello.

Le possibilità di cambiamento e miglioramento sono innumerevoli e solo una

minima parte di queste è stata testata, nonostante ciò i risultati sono comunque incoraggianti, aprendo nuove strade per l'AI Neurosimbolica.

A prescindere dalla reale validità e possibilità di applicazione del nuovo metodo, il risultato fondamentale da mettere in risalto è che l'iniezione di conoscenza può essere concretamente utilizzata per migliorare le performance e la robustezza dei modelli non educati.

Questi risultati, inoltre, dimostrano come ci sia sempre una strada alternativa, e come spesso la strada non debba necessariamente essere del tutto inesplorata. Spesso la soluzione ad un nostro problema è già in mezzo a noi in contesti insospettabili. Spesso è lei a trovarci, ma dobbiamo poi essere in grado di riconoscerla, riadattarla ed integrarla in modo proficuo.

Bibliografia

[1] M. Roberts, et al., "Common pitfalls and recommendations for using machine learning to detect and prognosticate for covid-19 using chest radiographs and ct scans", *Nature Machine Intelligence*, vol. 3, no. 3, pp. 199–217, 2021. [Online]. Available: https://doi.org/10.1038/s42256-021-00307-0.

[2] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition", *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37, pp. 1–10, 2015. [Online]. Available: https://www.cs.cmu.edu/ rsalakhu/papers/oneshot1.pdf.

[3] C. Trotter, "How to train your siamese neural network", *Towards Data Science*, 2020. [Online]. Available: https://towardsdatascience.com/how-to-train-your-siamese-neural-network-4c6da3259463.

[4] L. von Rueden et al., "Informed machine learning – a taxonomy and survey of integrating prior knowledge into learning systems," *IEEE* Transactions on Knowledge and Data Engineering, vol. 35, no. 1, pp. 614–633, 2023

[5] F. Leiser, S. Rank, M. Schmidt-Kraepelin, S. Thiebes, and A. Sunyaev, "Medical informed machine learning: A scoping review and future research directions," Artificial Intelligence in Medicine, vol. 145, p. 102676, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0933365723001902

[6] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations," 2017. [Online]. Available: https://arxiv.org/abs/1711.10561

[7] A. Daw, A. Karpatne, W. Watkins, J. Read, and V. Kumar, "Physics-guided neural networks (pgnn): An application in lake temperature modeling," 2021. [Online]. Available: https://arxiv.org/abs/1710.11431

[8] M. Liu et al., "A translational perspective towards clinical ai fairness," npj Digital Medicine, vol. 6, no. 1, p. 172, 2023. [Online]. Available: https://doi.org/10.1038/s41746-023-00918-4

[9] L. Dai, R. Fang, H. Li, X. Hou, B. Sheng, Q. Wu, and W. Jia, "Clinical report guided retinal microaneurysm detection with multi-sieving deep learning," IEEE transactions on medical imaging, vol. 37, no. 5, pp. 1149–1161, 2018.

[10] E. L. Silva, A. F. Sampaio, L. F. Teixeira, and M. J. M. Vasconcelos, "Cervical cancer detection and classification in cytology images using a hybrid approach," in Advances in Visual Computing: 16th International Symposium, ISVC 2021, Virtual Event, October 4-6, 2021, Proceedings, Part II. Springer, 2021, pp. 299–312.

[11] J. I. Orlando, E. Prokofyeva, M. Del Fresno, and M. B. Blaschko, "An ensemble deep learning based approach for red lesion detection in fundus images," Computer methods and programs in biomedicine, vol. 153, pp. 115–127, 2018.

[12] J. Huang, H. Yan, J. Li, H. M. Stewart, and F. Setzer, "Combining anatomical constraints and deep learning for 3-d cbct dental image multi-label segmentation," in 2021 IEEE 37th International Conference on Data Engineering (ICDE). IEEE, 2021, pp. 2750–2755.

[13] Y. Guo, G. Bai, and Y. Hu, "Using bayes network for prediction of type-2 diabetes," in 2012 International conference for internet technology and secured transactions. IEEE, 2012, pp. 471–472.

[14] P. K. Jain and O. P. Mahela, "Automatic analysis of the heart sound signal to build smart healthcare system," in Internet of Multimedia Things (IoMT), ser. Intelligent Data Centric Systems. Cambridge, Massachusetts: Elsevier, 2022, pp. 151–188.

[15] Z. Bouzid, Z. Faramand, R. E. Gregg, S. O. Frisch, C. Martin-Gill, S. Saba, C. Callaway, E. Sejdic, and S. Al-Zaiti, "In search of an ´ optimal subset of ecg features to augment the diagnosis of acute coronary syndrome at the emergency department," Journal of the American Heart Association, vol. 10, no. 3, p. e017871, 2021.

[16] G. Ciatto, F. Sabbatini, A. Agiollo, M. Magnini, and A. Omicini, "Symbolic knowledge extraction and injection with sub-symbolic predictors: A systematic literature review," ACM Computing Surveys, vol. 56, no. 6, pp. 161:1–161:35, Jun. 2024. [Online]. Available: https://dl.acm.org/doi/10.1145/3645103

[17] E. H. Shortliffe, "MYCIN: A rule-based computer program for advising physicians regarding antimicrobial therapy selection", *Proceedings of the American Federation for Clinical Research*, vol. 25, no. 2, pp. 19–21, 1977. [Online]. Available:

https://www.researchgate.net/publication/234816828_MYCIN_A_Rule-Based_Computer_Program_For_Advising_Physicians_Regarding_Antimicrobial_Therapy_Selection

[18] Towell, G. G., Shavlik, J. W. (1994). Knowledge-based artificial neural networks. Artificial Intelligence, 70(1-2), 119-165.

[19] Eleonora Giunchiglia, Mihaela Catalina Stoian, and Thomas Lukasiewicz. Deep learn- ing with logical constraints. arXiv preprint arXiv:2205.00523, 2022.

[20] Przemyslaw Biecek and Tomasz Burzykowski. *Explanatory model analysis: explore, explain, and examine predictive models*. Chapman and Hall/CRC, 2021.

[21] Andrea Rafanelli, Matteo Magnini, Andrea Agiollo, Giovanni Ciatto, and Andrea Omicini. *An empirical study on the robustness of knowledge injection techniques against data degradation*. Disponibile: https://cris.unibo.it/bitstream/11585/975934/1/paper_02.pdf

[22] Giovanni Ciatto, Federico Sabbatini, Andrea Agiollo, Matteo Magnini, and Andrea Omicini. Symbolic knowledge extraction and injection with sub-symbolic predictors: A systematic literature review. ACM Computing Surveys, 56(6), mar 2024.

[23] Matteo Magnini, Giovanni Ciatto, and Andrea Omicini. KINS: Knowledge Injection via Network Structuring. *Preprint*, 2024.

[24] Samy Badreddine, Artur d'Avila Garcez, Luciano Serafini, and Michael Spranger. Logic Tensor Networks. *arXiv preprint arXiv:2012.13635*, 2021.

[25] UCI Machine Learning Repository, "Pima Indians Diabetes Database," Kaggle, 2016. [Online]. Disponibile: https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database

[26] Kunapuli, G., Bennett, K.P., Shabbeer, A., Maclin, R., Shavlik, J. (2010). Online Knowledge-Based Support Vector Machines. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds) Machine Learning and Knowledge Discovery in Databases. ECML PKDD 2010. Lecture Notes in Computer Science(), vol 6322. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-15883-4_10

[27] W. Duch, R. Adamczak, K. Grabczewski, A new methodology of extraction, optimization and application of crisp and fuzzy logical rules, IEEE Transactions on Neural Networks 12 (2001) 277–306. URL: https://doi.org/10.1109/72.914524. doi:10.1109/72.914524.

[28] Y. Hayashi, S. Nakano, Use of a recursive-rule extraction algorithm with J48graft to achieve highly accurate and concise rule extraction from a large breast cancer dataset, Informatics in Medicine Unlocked 1 (2015) 9–16.

[29] W. Wolberg. "Breast Cancer Wisconsin (Original)," UCI Machine Learning Repository, 1990. [Online]. Disponibile: https://doi.org/10.24432/C5HP4Z.

Ringraziamenti

La conclusione di questa tesi rappresenta per me un traguardo importante: la chiusura di una fase fondamentale della mia vita, in cui ha avuto luogo la mia formazione personale e professionale, che mi permetterà di affrontare al meglio le sfide future. Tutto questo è stato possibile grazie al sostegno, alla guida e alla presenza costante di persone a cui dedico queste righe di sincera gratitudine.

Un ringraziamento speciale va alla mia relatrice, prof.ssa Sara Montagna, e alla correlatrice dott.ssa Christel Sirocchi, per la disponibilità e la professionalità con cui mi hanno accompagnato in questo lavoro. Il loro supporto è stato fondamentale, guidandomi con competenza e contribuendo in modo significativo alla mia crescita personale e accademica. Ringrazio Matteo Magnini per aver messo a disposizione PSyKI, per il testing dei metodi di iniezione presentati in questo lavoro.

Desidero ringraziare di cuore i miei genitori, per essere stati la mia base sicura in ogni fase di questo cammino. Grazie per aver creduto in me, per avermi dato fiducia, libertà e al contempo orientamento, e per avermi sempre sostenuto con amore e pazienza, anche nei momenti più difficili. Ringrazio mio padre, oltre che per il sostegno economico, per la tranquillità e serenità trasmessa in tutto il percorso di studi. Ringrazio mia madre, per la sua presenza costante e il supporto morale, spesso reciproco.

Ringrazio i miei amici, per aver rallegrato le mie giornate, regalandomi la giusta dose di spensieratezza e risate, indispensabili in periodi pieni ed impegnativi come quelli appena trascorsi.

Un ringraziamento profondo va infine a Rossella, la mia fidanzata, per aver condiviso con me gioie e fatiche, per il suo supporto incondizionato e la sua vicinanza costante. La sua presenza ha reso questo percorso meno tortuoso e infinitamente più ricco e significativo.

Qui avviene il passaggio ad un nuovo capitolo, ancora da esplorare e vivere. Tanta la voglia di mettermi in gioco, conoscere ed imparare, mettendo a disposizione le mie abilità e conoscenze professionali.