Università degli Studi di Bari Dipartimento di Informatica

Dottorato in Informatica e Matematica - XXX Ciclo

Apprendimento di Modelli Grafici (Probabilistici) per il Representation Learning

Proposta di Progetto di Ricerca

Dottorando: Antonio Vergari
Tutor: Prof.ssa Floriana Esposito
Co-Tutor: Dott. Nicola Di Mauro
Coordinatore: Prof. Donato Malerba

1. Obiettivi e panoramica

L'obiettivo di costruire sistemi che inducano modelli a partire da dati osservati è proprio del ***Machine Learning*** (**ML**) [7] (o Apprendimento Automatico, in italiano) e del più specifico ***Statistical Relational Learning*** (**SRL**) [24] quando vi è la necessità di gestire l'incertezza di dati appartenenti a domini strutturati. La proposta di questo progetto prevede di studiare come apprendere sia i modelli che abbiano generato i dati in questi constesti complessi, *sia, allo stesso tempo*, una rappresentazione, latente e significativa, per i dati stessi. Per tale ragione lo si può inquadrare nel contesto del ***Representation Learning*** (**RL**) [3], il quale, quando le rappresentazioni indotte sono stratificate in differenti livelli di astrazione, prende il nome di ***Deep Learning*** (**DL**).

Scendendo ancor più nello specifico, l'obiettivo diviene l'investigare come ***Modelli Grafici (Probabilistici)*** (**(P)GM**) [36] *possono essere impiegati per il RL e DL*, ossia studiare come apprendere rappresentazioni latenti strutturate in ambiti dove estrarre automaticamente *embedding* (spazi di rappresentazione) complessi dai dati ed estrapolare l'interazione fra le variabili è fondamentale.

Se in possesso di una architettura *deep*, i cui parametri sono stati appresi dai dati, si è in grado di estrarre nuove **rappresentazioni** per le feature in input interpretando i livelli che contengono le feature **latenti** [37, 66]. La loro utilità è ben nota in numerosi campi e contesti applicativi, anche quando il loro impiego avviene in task successivi, partendo dall'NLP ed arrivando all'Image Processing [6, 2] e la loro risonanza negli ultimi anni è enorme. Tuttavia, non è tutto oro ciò che luccica, la profondità delle architetture può essere la causa di intrattabilità e colli di bottiglia nelle fasi di apprendimento [3] e, spesso, le relazioni estratte fra le nuove rappresentazioni per le feature sono oscure e dunque non facilmente interpretabili o riusabili.

Considerando che ottenere tali rappresentazioni equivale ad elicitare le interazioni latenti presenti fra i dati, e che la struttura di una architettura collegante le variabili nascoste (hidden) altro non rappresenta che le relazioni che insistono fra di esse, si potrebbe essere in grado di raggiungere obiettivi similari cercando di apprendere esplicitamente un modello grafico (probabilistico) che tenga in conto della presenza di feature latenti a priori o addirittura provando ad indurre la migliore architettura che mostri tali relazioni fra feature, ovvero, **apprendere la struttura** stessa di una architettura deep [58, 48].

Modelli e metodologie esistenti verranno studiate nell'ottica di capirne punti di forza e debolezze, mentre estensioni a questi, o modelli interamente nuovi saranno proposti e confrontati con i primi (ad esempio si pensi ad estensioni a formalismi relazionali).

2. Contesto e motivazione

La classica pipeline per un passo di apprendimento automatico in un dominio generico e complesso prevederebbe un passo di *feature selection* o *extraction* fra quelli iniziali[2, 4]. Task che dipendono fortemente dalle feature utilizzate, come image classification e speech recognition vedono la stragrande maggioranza degli sforzi compiuti (stimata in $\sim 90\%$ dei costi complessivi [37]) in processi di feature engineering; i ragguardevoli traguardi raggiunti nelle passate decadi in questi ambiti sono dovuti non in poca parte a feature ad-hoc, ritagliate su misura per problemi e domini molto specifici, come SIFT, MFCC e via discorrendo [32]. Questo "overtuning"

costituisce un collo di bottiglia e in termini di potere di generalizzazione dei modelli appresi, e in termini di tempo speso ad estratte e testare descrizioni dei dati.

A partire dalla loro prima introduzione nel 2006 [5], architetture deep come ***Stacked (Denoising) Autoencoders*** (S(d)A) [60, 61] e ***Deep Belief Networks*** (DBN) [31, 33] hanno dimostrato come è possibile costruire modelli (in maniera supervisionata) altamente accurati senza partire da feature estratte appositamente, e tuttavia giungendo a rappresentazioni latenti di rilievo. Il significato che queste rappresentazioni portano con sé, ad esempio, su dataset di immagini standard può essere associato (anche dall'occhio umano) alla presenza di semplici filtri d'immagine, organizzati in una altrettanto semplice gerarchia: line detector che diventano edge detector che diventano pattern di parti riconosciute [3]. Nonostante si possa far risalire i primi algoritmi di RL fino alle introduzioni dei livelli hidden nelle reti neurali (Multi-Layer Perceptron [54]), l'introduzione di questi layer con il solo obiettivo di migliorare le performance di un classificatore non equivale ad elicitare le interazioni non osservate a partire da quelle che lo sono; affermare, semplificando, che DL o RL si riducono ad apprendere (quelli che spesso sono unicamente i parametri di) una rete neurale è *profondamente* fallace e fa perdere l'obiettivo di fondo, la scoperta di rappresentazioni ricche e latenti che siano utili anche in più task. Per esserlo, di fatto, esse devono possedere delle caratteristiche peculiari.

2.1. **Rappresentazioni *deep*.** Per modellare efficacemente *funzioni altamente varianti* (high varying functions)[1] cercando di essere il più efficienti possibile (ad esempio mantenendo uno spazio dei parametri rappresentabile in forma compatta) e cercando di non overfittare sui soli dati disponibili, una buona rappresentazione deep dovrebbe essere [4]:

- **distribuita:** (distributed) poiché le variabili osservate sono il risultato di un processo latente che combina più di un fattore nascosto;
- **sparsa:** (sparse) dato che questo processo latente coinvolge, per ogni variabile prodotta, solo una piccola porzione dei fattori introdotti;
- **invariante a trasformazioni affini:** nel senso che il processo latente è in grado di generare le feature osservate tramite l'applicazione di trasformazioni (quali le trasformazioni geometriche affini come *rototraslazioni* e *scaling* per le immagini o, ancora, *traslitterazioni di contesto* per le accezioni delle parole) da cui i fattori latenti dovrebbero essere invarianti;
- **rappresentante della reale manifold dei dati:** (*manifold hypothesis* [4]) il che vuol dire che è in grado di descrivere la porzione significativa, anche se molto piccola, della grande manifold su cui giacciono le istanze osservate.

Tecniche e modelli ben noti come Principal/Independent Component Analysis (P/ICA) [30], (Nonnegative) Matrix Factorization ((N)MF) [12], e Semantiche Distribuzionali (Distributional Semantics) [17] in NLP ricercano trasformazioni dello spazio delle feature che hanno in comune alcune di, se non tutte, queste proprietà e sono, in questo senso, affini al RL, tuttavia nessun ulteriore utilizzo, che non sia l'incorporamento di queste trasformazioni in un classificatore, viene impiegato. L'ulteriore vantaggio proposto dal DL è quello di sbrogliare le relazioni latenti a

[1]La vera maledizione risiede nell'alto numero di variazioni che una funzione, intesa come il modello reale sottostante la generazione dei dati, possiede, piuttosto che nel numero delle dimensioni (feature) che compongono il piano in cui la si rappresenta.Si immagini una curva con molteplici massimi/minimi locali in uno spazio bidimensionale o proiettata in uno spazio a più dimensioni.

differenti livelli di granularità, proprio costruendo più di un livello di rappresentazioni di questo tipo[2].

Addestrare architetture composte di numerosi layer è un task tutt'altro che banale. La problematica, probabilmente più frequente, da affrontare riguarda l'efficienza nel passo di inferenza (che di conseguenza si ripercuote in tutte le fasi dell'apprendimento che fanno uso di routine per fare inferenza), o per il vasto numero di parametri coinvolti o per gli effetti che si riscontrano quando si propagano gli aggiornamenti dei parametri durante l'apprendimento attraverso tutti i layer delle architetture (fra questi c'è il fenomeno della *diffusione del gradiente* (gradient diluition) quando si applicano tecniche di ottimizzazione come la discesa del gradiente) [33]. Nell'affrontare entrambe, diversi approcci di apprendimento **non supervisionato** possono venire in soccorso come fasi di ***pretraining*** per estrarre in maniera robusta relazioni fra le rappresentazioni latenti, in maniera tale da favorire la successiva fase di apprendimento supervisionato facendola partire da un conveniente posizionamento nella manifold dei parametri [32, 3].

Anche con l'aggiunta di tali "espedienti", la trattabilità, qui intesa come complessità al più polinomiale (nello spazio delle feature di input) dell'inferenza, non è affatto garantita. Una delle ragioni dietro ciò risiede nella costruzione delle architetture stesse: nonostante esse non facciano affidamento ad estrattori di feature fissati a priori, nella stragrande maggioranza dei casi la struttura delle reti deep *è fissata* a priori, e coinvolge la completa connessione dei nodi presenti nei layer adiacenti (lo spazio dei parametri è il più ampio possibile, a prescindere dalla struttura della manifold sottostante). Se il processo di costruzione della architettura potesse essere automatizzato a partire dai dati, allora si sarebbe in grado di imporre dei vincoli sulla struttura da imparare limitando potenzialmente sì l'espressività ma ricercando la trattabilità. Questo modo di ragionare non è dissimile dal proporre ciò che è stato già fatto per l'estrazione delle rappresentazioni latenti, ma ad un più alto livello di astrazione: cioè considerando stavolta le relazioni fra di esse, nella forma della struttura della rete in cui compaiono.

2.2. **Inferenza trattabile e apprendimento di strutture.** I PGM sono in grado di rappresentare in forma compatta distribuzioni congiunte di probabilità su un insieme di variabili (aleatorie, v.a. da qui in poi) tramite rappresentazioni a grafo i cui nodi corrispondono alle v.a. osservate o no (in caso di v.a. latenti introdotte a priori) ed i cui archi denotano le relazioni dipendenza fra di queste [36]. Non tutte le architetture deep possono essere considerate probabilistiche, alcune di quelle che lo sono possono essere considerate PGM, mentre altre possono essere *costruite* a partire da modelli grafici (ad esempio nella fase di pretraining) semplici come le ***Restricted Boltzmann Machines*** (RBM) [38, 3]; dall'altra parte dell'equazione, i modelli grafici generativi che mostrano dipendenze fra variabili strutturate in livelli possono essere interpretati in chiave deep, per esempio i *plate models* mutuati dai metodi ***Bayesiani Non Parametrici*** (Bayesian Nonparametrics, BNP) [35, 34] anche se il numero dei loro livelli risulta essere piuttosto piccolo, e l'attenzione in

[2]Da questa prospettiva, molti algoritmi comuni in ML possono essere reinterpretati come architetture *piatte* (shallow) per il RL [3], per esempio, si consideri una SVM kernel non lineare, il primo livello di una architettura derivante da essa trasporta lo spazio in input nello spazio del kernel mentre il secondo, ed ultimo, livello applica un discriminatore lineare alla precedente rappresentazione. Come già detto, questo non qualifica in automatico lo spazio delle feature di input proiettate come una rappresentazione utile appresa.

quel contesto non è incentrata sulle rappresentazioni latenti se non all'interno del framework generativo.

Algoritmi per l'apprendimento della struttura di un PGM sono stati applicati ad una gran varietà di task con lo scopo di portare alla luce le dipendenze nascoste dei dati [22, 15, 28]. Ad alto livello, questa ricerca viene eseguita come un problema di ricerca globale nello spazio delle possibili strutture, oppure in maniera costruttiva, e spesso greedy, combinando modelli appresi localmente [29], ed in entrambi i casi la maggior causa di inefficienze è fare inferenza nei modelli appresi. Per questi casi il problema della trattabilità dell'inferenza risiede nel computo della *funzione di partizione*, la quale opera, in generale, su un numero esponenzialmente grande di termini [14, 36].

Questa problematica è stata affrontata negli anni da diverse prospettive: con routine sempre più efficienti per fare *inferenza approssimata*[3] e con l'introduzione di **modelli trattabili** che, sacrificando l'espressività di PGM maggiormente complessi, garantiscono una complessità polinomiale e forniscono procedure e routine per l'inferenza efficienti in teoria e in pratica [10, 41]. Un terzo, più recente, approccio consiste nel cercare di fare *lifted inference* [59], ovvero di cercare di trarre vantaggio dalla struttura del modello per astrarre ed aggregare v.a. con l'obiettivo di abbassare il numero di termini nella funzione di partizione fino a giungere a valutazioni di sotto strutture dalla complessità polinomiale.

Sempre di recente, le **Sum-Product Networks** (SPN) [49] sono state presentate come un PGM che promette *inferenza esatta ed allo stesso tempo trattabile*, costituendo per altro una architettura deep poiché alternano livelli di v.a. latenti rappresentanti misture o fattorizzazioni non osservabili direttamente nei dati. Le SPN hanno ravvivato l'interesse per entrambe le comunità, DL/RL e PGM, grazie a queste particolari proprietà, di fatto aprendo alla possibilità di apprendere la struttura di un modello grafico anche dalla prospettiva del DL, con l'estrazione di feature latenti probabilistiche.

In conclusione, l'opportunità di investigare la possibilità di indurre direttamente una struttura dai dati nella forma di architetture deep nasce come una necessità da ambo le comunità e l'interesse in materia non fa che essere corroborato dal successo che entrambi i filoni di ricerca ottengono, ognuno sul proprio fronte.

3. Stato dell'arte

In questa sezione forniremo una generale, e tuttavia comprensiva, panoramica su come le architetture introdotte vengono impiegate nella ricerca attuale e su quali problemi vengono affrontati nelle comunità che attualmente lavorano su questi temi producendo i risultati più all'avanguardia; si spazierà da aree di ricerca differenti, per vedere particolari applicazioni per alcune di esse si faccia riferimento alla sezione successiva.

Alcuni rilevanti contributi sono giunti dalla comunità dei BNP a partire dalla ricerca sulle migliori strutture generanti gli esempi osservati, siano in forma di grafo, di albero o di una semplice catena (la topologia di una struttura non viene fissata a priori, ma appresa dai dati) [34], qui lo sforzo maggiore è speso nel modellare l'apprendimento del processo di apprendimento attraverso la rappresentazione di un modello generativo. Le descrizioni estratte sono di fatto strutture ed indicano

[3] A volte, quando approssimare l'inferenza è obbligatorio, le routine "efficienti" non sono trattabili.

l'interazione fra le variabili osservate, tuttavia gli algoritmi proposti per apprenderle sono altamente inefficienti, basati su routine MCMC per fare inferenza.

Modelli probabilistici trattabili che non siano SPN ricevono lo stesso attenzione, anche se in maniera minore poiché meno generali [48]: i ***Thin Junction Trees*** [10], le ***Multi Linear Representations*** [53], Hinge-loss Markov Random Fields [1] e gli ***Arithmetic Circuits*** (AC), i quali sono stati introdotti per "compilare" in modelli trattabili le Bayesian Networks (BN) [14] e dai quali le SPN sono state derivate. Una specializzazione di una BN (ossia introducendo vincoli sul modello, sempre limitando l'espressività in favore della trattabilità) meno recente ma ancora ricercata sono gli ***alberi di Chow-Liu*** [11] in cui le dipendenze condizionali esprimibili ammettono che una v.a. possa avere un solo genitore nel grafo.

In aggiunta a questi, PGM molto semplici come Restricted Boltzmann Machines sono stati con successo adoperati in DL con l'introduzione di una procedura maggiormente efficiente per fare inferenza, l'ottimizzazione della *contrastive divergence* in luogo della classica funzione di likelihood a partire dai dati [3]. Essi costituiscono i mattoni con cui è possibile costruire molteplici architetture deep, e tuttavia compaiono in letteratura anche utilizzati singolarmente: dal loro utilizzo standalone alcuni interessanti modelli relazioni sono stati proposti, o aggiungendo un layer di input che agisca da proposizionalizzatore traducendo formule logiche in feature dell'ordine zero [39] o cercando di tradurre le attivazioni osservate dei nodi nascosti (ed i relativi path) in una qualche formulazione di logica temporale [16].

Subito dopo la loro comparsa in letteratura, le SPN sono state estensivamente ricercate con l'obiettivo di apprenderne la struttura direttamente dai dati; ad oggi, differenti algoritmi sono stati proposti, ma tutti condividono una aspetto comune: l'utilizzo di tecniche non supervisionate, una su tutte il clustering, per la elicitazione delle interazioni fra le feature latenti. Il primo, in ordine cronologico, in [19], usa un semplice *K-Means* sulle variabili osservate (non sulle istanze) procedendo top-down ricorsivamente; in maniera opposta, la versione proposta in [46] cerca di decomporre il problema muovendosi bottom-up per mezzo di fusioni successive di variabili utilizzando come procedure di scelta procedure (test statistici) basate sulla *Informazione Mutua*. La prima procedura legittimata teoricamente viene presentata in [26] dove è la matrice dei dati ad essere iterativamente partizionata lungo righe e colonne in accordo a qualche criterio di similarità o indipendenza (quella proposta è una famiglia di algoritmi), con la risultante prodotta nella forma di una SPN ad albero. L'algoritmo che attualmente ottiene le performance migliori è dovuto a [52], dove la struttura appresa non è una semplice SPN ma una "SPN di modelli trattabili" ovvero le cui foglie possono rappresentare altre approssimazioni di probabilità. Una altra variante degna di nota viene proposta in [47] dove un modello leggermente meno espressivo, le *Selective SPN*, viene appreso attraverso una Stochastic Local Search in grando di garantire un livello di efficienza attraverso la formulazione della funzione di costo in forma chiusa, nonostante la limitatezza espressiva rispetto gli altri modelli, le performance sono confrontabili con gli altri algoritmi finora elencati su un insiemen di dataset standard.

L'idea di estendere le SPN, e modelli grafici a loro affini, verso formalismi logici per trattare dati multi relazionali è stata esplorata tentativamente da Pedro Domingos in più di un lavoro. In [20] viene introdotta una versione "ridotta" delle *Markov Logic* limitandone l'espressività con dei vincoli similari alle *Description Logic* per tradurre una gerarchia *part-based* in una SPN con l'obiettivo di fare inferenza in maniera

efficiente. Una prima estensione di questo modello appare in [63]: il linguaggio di rappresentazione per la logica sottostante viene esteso per includere costrutti maggiormente espressivi capaci di indicare un maggior numero di relazioni, ma ma è sempre una SPN ad essere introdotta come via per compilare la sottostante base di conoscenza, non apprendibile dai dati. Il più recente raffinamento appare nella forma delle ***Relational Sum Product Network*** in [44] in cui l'introduzione del concetto di *exchangeblility* fra gli esempi aiuta a generalizzare le proprietà delle classi utilizzate nei precedenti tentativi, riuscendo a mantenere la trattabilità sempre grazie ad una SPN.

Altri modelli grafici che garantiscono la trattabilità hanno anch'essi ricevuto il loro slot di attenzione nella ricerca sull'apprendimento delle strutture; i Markov Random Field, pur non essendo di per sé trattabili in generale, sono stati con successo appresi attraverso la generazione di sottospazi di plausibili feature latenti in maniera aleatoria in [29]; gli Arithmetic Circuit si sono rivelati per fare inferenza al pari delle SPN, e sono stati impiegati per cogliere le interazioni non dirette nelle Markov Network [28]; altri approcci analoghi hanno sfruttato gli alberi di decisione [40] anche se, una volta appreso un modello grafico complesso in questa maniera, non persiste alcun vantaggio nel fare nuove inferenze con la struttura ottenuta rispetto a tutti gli algoritmi che danno in output un modello trattabile. Un semplice quanto intelligente stratagemma prevede la costruzione di una mistura di AC a partire dalle partizioni ottenute a seguito di un passo di clustering sull'insieme originario delle istanze (i cui coefficienti vengono stimati attraverso l'impiego di EM) [51]. Apprendere un albero OR al di sopra di una versione degli alberi di Chow-Liu viene presentata in [50] dove vengono mostrati i migliori, ad ora, tempi di apprendimento fornendo al contempo anche una struttura trattabile in uscita.

Circa l'apprendere rappresentazioni nascoste e deep da strutture fissate a priori, ovvero indurre a partire dai dati solo i valori per i parametri che governano le reti, i lavori presentati fino al 2009 sono sufficientemente trattati a fondo in [3] e coinvolgono tutti, sostanzialmente, la ingegnerizzazione di passi di *ottimizzazione numerica* per la minimizzazione vincolata o no di funzioni di costo sui parametri; una altra panoramica altrettanto comprensiva di lavori qui già citati o meno è presente in [18]; un altro pattern frequente riscontrabile in questi lavori è l'introduzione della fase di pretraining nella forma di un passo di apprendimento supervisionato che consiste nell'apprendimento greedy di un livello alla volta, partendo dal più basso, quello di input, la presenza di esempi non etichettati aiuta a stimare la distribuzione congiunta sottostante i dati in maniera decisamente più robusta rispetto a metodi che non impiegano questa fase. Degne di nota sono le reti molto profonde addestrate in [55] e le SPN poste al di sopra di una ***Convolutional Neural Network*** (CNN) per il riconoscimento di immagini in [25] dato lo sforzo impiegato dagli autori per minimizzare i problemi di diluizione del gradiente durante l'apprendimento. Come metodo per affrontare gli stessi problemi nell'apprendimento di strutture deep, fra le nuove proposte fra le regolarizzazioni di metodi iterativi di ottimizzazione, l'introduzione di **Dropout** in [56] promette di rendere la fase di pretraining inutile, ulteriori derivazioni a partire da questo si possono trovare in **DropConnect** [62] e **DropAll** [21], l'idea portante alla base di tutti e tre è di deattivare alcuni nodi o archi all'interno di un livello mentre si apprendono i pesi per rafforzare le struttura rispetto piccole variazioni (corruzioni) presenti nell'input, in una maniera del tutto simile al mediare il valore ottenuto da un ensemble di modelli probabilistici.

Alcuni tentativi, decisamente interessanti, sono stati mossi per apprendere rappresentazioni utili a partire da dati strutturati se non interamente relazionali. Uno fra i più recenti, che qui si cita, viene presentato in [9] come un metodo per generalizzare una CNN sopra esempi in forma di grafo attraverso la derivazione di una generalizzazione per l'operatore di convoluzione nel dominio spettrale. Un altro lavoro su una generalizzazione teorica di questo stesso tipo di rappresentazioni strutturate latenti può essere trovato in [13] dove le operazioni ammesse per i livelli della struttura possono essere quelle considerate da un gruppo di simmetria, in tal modo allargando lo spazio delle rappresentazioni trovabili nella manifold dei dati. Essere in grado di sfruttare e rappresentare le simmetrie diviene di grande interesse anche per la comunità dei PGM, come dimostra il recente lavoro di [27] in cui lo scopo è sempre apprendere un modello grafico deep che tenga in considerazione le possibili trasformazioni affini per la scoperta di fattori ad esse invarianti.

Dal lato del RL, l'apprendimento di embedding utili, ma non rientranti in una gerarchia esplicita, costituisce il tema di fondo dell'apprendimento dei modelli energetici, ***Energy Model learning*** [37]; il loro principale vantaggio è quello di consentire formulazioni compatte e pulite di funzioni di Energia che godono di diverse proprietà (derivabilità, convergenza, etc) per incapsulare gli embedding nella forma di spazi vettoriali; il concetto cardine dietro queste rappresentazioni è apprendere spazi nei quali le formulazioni energetiche associano valori bassi a configurazioni indicanti gli esempi positivi o semplicemente visti nella manifold, alti nelle altre regioni dello spazio. Essi sono in qualche modo derivazioni o generalizzazioni delle formulazioni energetiche (e probabilistiche) riscontrate nella letteratura sui PGM (a loro volta mutuati dalla letteratura sulla meccanica statistica). Dalla loro proposta diverse estensioni verso architetture deep sono state avanzate, come quella mostrata in [45].

3.1. **Problemi aperti e sfide.** Per citare John Langford che cita Geoff Hinton: *"The great thing about deep belief networks is that they work"*. Questa frase ben riassume la diffusa attitudine comunemente rivolta a DL e RL oggigiorno: una corsa verso modelli altamente accurati e performanti (non nel senso di efficienza) che però non sono ancora pienamente compresi.

Apprendere i parametri di una architettura deep risulta in una gerarchia di rappresentazioni indotte, come già ricordato più volte in questo documento, tuttavia non c'è ancora un automatismo (nel senso di metodologia motivata in ML) per tradurre o anche solo associare queste a rappresentazioni simboliche (se non logiche). L'interpretabilità che a loro si dà attraverso la vista di noi umani appare essere 'sufficiente' fintantoché si è in grado di produrre risultati allo stato dell'arte. Un metodo che fosse in grado di collegare rappresentazioni simbolica ad embedding latenti aprirebbe da solo un intero filone di ricerca, che prenderebbe il nome dello spesso invocato ***Deep Relational Learning***. Ricercare qualcosa del genere è così ambizioso che va al di là dello scopo di questo progetto di ricerca, e tuttavia l'indagine sulla interpretabilità degli embedding in chiave relazionale potrebbe essere una direzione lungo la quale indirizzare i propri sforzi.

In maniera altrettanto simmetrica, poco interesse è stato ancora mostrato nel cercare di interpretare o anche descrivere le rappresentazioni hidden che vengono apprense con le architetture deep dal lato dei PGM. Per le SPN una spiegazione tentativa è stata avanzata in [48, 25] nella forma di decomposizione in parti e misture dell'input, ma neanche una visualizzazione a supporto di ciò è stata avanzata.

Altre grandi sfide riguardano le implementazioni degli algoritmi per i modelli finora citati, anche se la trattabilità per le routine di inferenza può essere garantita teoricamente, la complessità cubica, e quindi polinomiale, che alcuni algoritmi mantengono, in presenza di dataset non giocattolo, richiede implementazioni iper efficienti e di parallelismo massivo per scalare. Indagare come implementare versioni note di questi algoritmi per architetture come la GPU non è cosa immediata, ma segna la strada di ricerca che viene regolarmente intrapresa dai grandi gruppi di ricerca.

3.2. **Comunità attive.** Il vibrante e crescente interesse per queste tematiche nelle diverse comunità di apprendimento automatico può essere facilmente valutato dalla comparsa di conferenze a tema, workshop e laboratori. La prima fra queste ad ambito internazionale è stata la ***International Conference for Learning Representations*** (**ICLR**), giunta alla sua terza installazione quest'anno. Workshop a tema DL compaiono con frequenza in tutte le maggiori conferenze di ML, ad esempio da tre edizio di ICML `http://www.uncg.edu/cmp/icml/`, o nella forma di sessioni di tutorial sulle tecnologie necessarie per affrontare siffatte problematiche, o ancora sotto forma di competizioni `http://deeplearning.net/icml2013-workshop-competition/`.

L'attenzione per i modelli trattabili cresce anch'essa di un fattore comparabile, e la convergenza fra gli interessi delle due comunità si avvicina. A dimostrazione di ciò ad ICML2014, ***Learning Tractable Probabilistic Models*** (**LTPM**) è stato il primo tentativo internazionale di mettere le due comunità intorno ad uno stesso tavolo `http://ltpm2014.cs.washington.edu/sites.google.com/site/ltpm2014/index.html`.

3.3. **Referenti esterni.** Qui si elencano dei possibili contatti, fra dei profili interessanti e di rilievo per la linea di ricerca finora delineata (potrebbero essere approcciati in futuro come collaboratori, revisori esterni o membri del comitato di valutazione dell'elaborato finale):

- Pedro Domingos - dall'University of Washington, a capo della ricerca su SPNs e modelli trattabili in generale `http://homes.cs.washington.edu/~pedrod/`
- Fabrizio Costa - direttore del Department of Bioinformatics of the University of Freiburg, facente ricerca sui PGM per l'apprendimento di struttura e parametri, con un occhio particolare per le applicazioni in Bioinformatica `http://www.bioinf.uni-freiburg.de/~costa/`
- Robert Peharz - del Signal Processing Laboratory della Graz University of Technology di Graz, autore di due lavori sull'apprendimento della struttura di una SPN `https://www.spsc.tugraz.at/people/robert-peharz`
- Giorgio Corani - senior researcher all' IDSIA, `http://people.idsia.ch/~giorgio/` e membro organizzatore del workshop internazionale sui PGM `http://ii.tudelft.nl/bnvki/?p=713`

4. Applicazioni

Qui si delineano non le precise aree in cui proporre applicazioni per il progetto di ricerca, quanto esempi di impiego dei metodi e modelli introdotti estremamente all'avanguardia, cercando di mostrate le grandi potenzialità di tali metodi.

Una delle maggiori per interesse ed impiego, se non la maggiore, delle applicazioni per DL e RL è il riconoscimento di immagini. In questo contesto le rappresentazioni

latenti sono altamente sfruttabili per apprendere classificatori robusti rispetto ad un elevato numero di trasformazioni.

Come già citato, l'allargamento dello spazio dei parametri e l'impilare livelli su livelli nelle architetture deep non implica una facile fase di apprendimento, anche se, quando in presenza di enormi risorse computazionali come in Google o Baidu, e con intelligenti ottimizzazioni per GPU o altrettanto furbe tecniche di data augmentation si è *facilmente* in grado di battere ripetutamente i migliori punteggi in task rinomatamente difficili come riconoscimento di entità su dataset di milioni di immagini e decine di migliaia di classi. Uno dei primi lavori, ad approccio brute force, in tal senso può essere ricercato in [23], successivamente architetture maggiormente robuste e più accurate sono state sviluppate in [57] solo per essere, ancor più recentemente, battute in [65].

Una similare fortuna ha riguardato anche il campo del NLP dove gli embedding nascosti di parole, significati, frasi, paragrafi di testo e così via sono stati sfruttati a fondo in architetture deep. Un caso notevole e recente lo si può trovare in [64] in cui una rete deep mappa parole in lingua inglese in uno spazio semantico a poche dimensioni.

Nel mondo del Semantic Web, la ricerca di rappresentazioni latenti ha portato alla formulazione energetica del problema di ricerca degli embedding su rappresentazioni a grafo come possono essere quelle derivanti da formalismi come RGDF su differenti task come graph summarization, link prediction and probabilistic query ranking. Alcuni tentativi notevoli di raggiungere queste formulazioni, in architetture shallow, si possono trovare a partire da [8].

5. Fasi ed attività

Le attività cuore di questo progetto, non potranno che essere *ricerca*, *sviluppo* e *valutazione*.

Sostanzialmente, la proposta che qui si fa è quella di strutturare le attività in maniera tale che le fasi di ricerca vengano seguite immediatamente da fasi di sviluppo per saggiare il prima possibile la bontà della ricerca prodotta fino a quel punto (valutazione anticipata).

Scendendo maggiormente nello specifico, l'attività di ricerca potrà essere così declinata:

- studio dello stato dell'arte nei campi del DL, RL e PGM, ossia sviluppo di una vasta conoscenza di base nei sopracitati campi studiando i lavori fondamentali in letteratura;
- formulazione di modalità per riversare la conoscenza acquisita (stato dell'arte) in progetti in fieri o nascenti, ossia proporre applicazioni;
- contaminazione con idee provenienti da altri ambiti e campi di ricerca (ad esempio revisionando lavori di ricerca di campi differenti ma potenzialmente affini);
- produzione di piccoli report a frequenza regolare sulla ricerca corrente, ovvero lasciare traccia delle strade intraprese in maniera incrementale;
- condivisione e circolo della conoscenza attraverso collaborazioni (ad esempio partecipando a summer schools, workshops, conferenze, ma anche potenziando la comunicazione all'interno del gruppo di ricerca o allargando questo);
- stesura del documento finale di tesi.

Ugualmente la fase di sviluppo può essere vista come l'insieme delle attività in cui:

- realizzare prototipi rapidi per la dimostrazione e condivisione delle idee generate nella fase precedente;
- collaborazioni su terreni comuni e pratici con altri ricercatori (per esempio la costruzione di e il contribuire a implementazioni comuni da diffondere nella comunità dei PGM);
- produzione della documentazione e dei proof of concept da affiancare alla stesura dei report.

In ultimo, le attività di valutazione si occuperanno di:

- progettazione degli esperimenti con cui testare i metodi ed i modelli sviluppati;
- analisi dei risultati delle sperimentazioni e riformulazione degli obiettivi di ricerca alla luce di questi;
- sviluppo della abilità di revisionare e giudicare il proprio lavoro e quello degli altri attraverso strumenti di peer review (ad esempio valutando i più notevoli lavori in ambiti di ricerca affini);
- condivisione dei risultati e ricerca di valutazione esterna, ovvero partecipazione a workshop, conferenze, come anche rendere il codice disponibile in rete.

5.1. **Schedulazione annuale.** Una suddivisione tentativa delle attività su scale annuale e quadrimestrale viene mostrata nel diagramma in figura 1. In esso le attività che non sono facilmente schedulabili in periodi limitati (gestione dei progetti, condivisione del lavoro di ricerca e partecipazione alla comunità scientifica) ricoprono un periodo più lungo e sono rappresentate da una texture tratteggiata.

Sebbene indicativa, la rappresentazione mostra come il primo anno sia maggiormente allocato alla ricerca sullo stato dell'arte, il secondo e il terzo alla implementazione e valutazione dei modelli proposti. Sono inoltre presenti le sequenze cicliche di prototipazione, design sperimentale ed analisi dei risultati, intervallate dai report di avanzamento periodici e da un ciclo di documentazione maggiormente fitto.

6. Valutazione del progetto

Ogniqualvolta un nuovo modello o estensione algoritmica viene proposto, la sua valutazione consisterà in una rigorosa sperimentazione su dataset reali in comparazione con le precedenti versioni e/o il corrente stato dell'arte. La definizione del processo sarà sicuramente influenzata dal task che ci si propone di affrontare e dal contesto applicativo di test, pertanto non è possibile, ora, definire a priori in maniera più specifica una metodologia che non sia quella classica del metodo scientifico. Di seguito si daranno i riferimenti per alcune delle possibili metriche e dei potenziali dataset utilizzabili in fase di sperimentazione.

6.1. **Metriche.** Il confronto fra i modelli proposti rispetto le controparti dello stato dell'arte potrebbe richiedere il coinvolgimento di differenti metriche di valutazione, la cui scelta verrà fatta in riferimento al particolare framework in cui si opererà, alle metriche precedentemente utilizzate nei lavori con cui ci si misura ed ai punti di forza del metodo proposto che è necessario evidenziare. Come esempio, nel framework generativo, poiché l'interesse è quello di valutare quanto un modello appreso sia in grado di generare le istanze di un insieme di test a partire dalla distribuzione degli

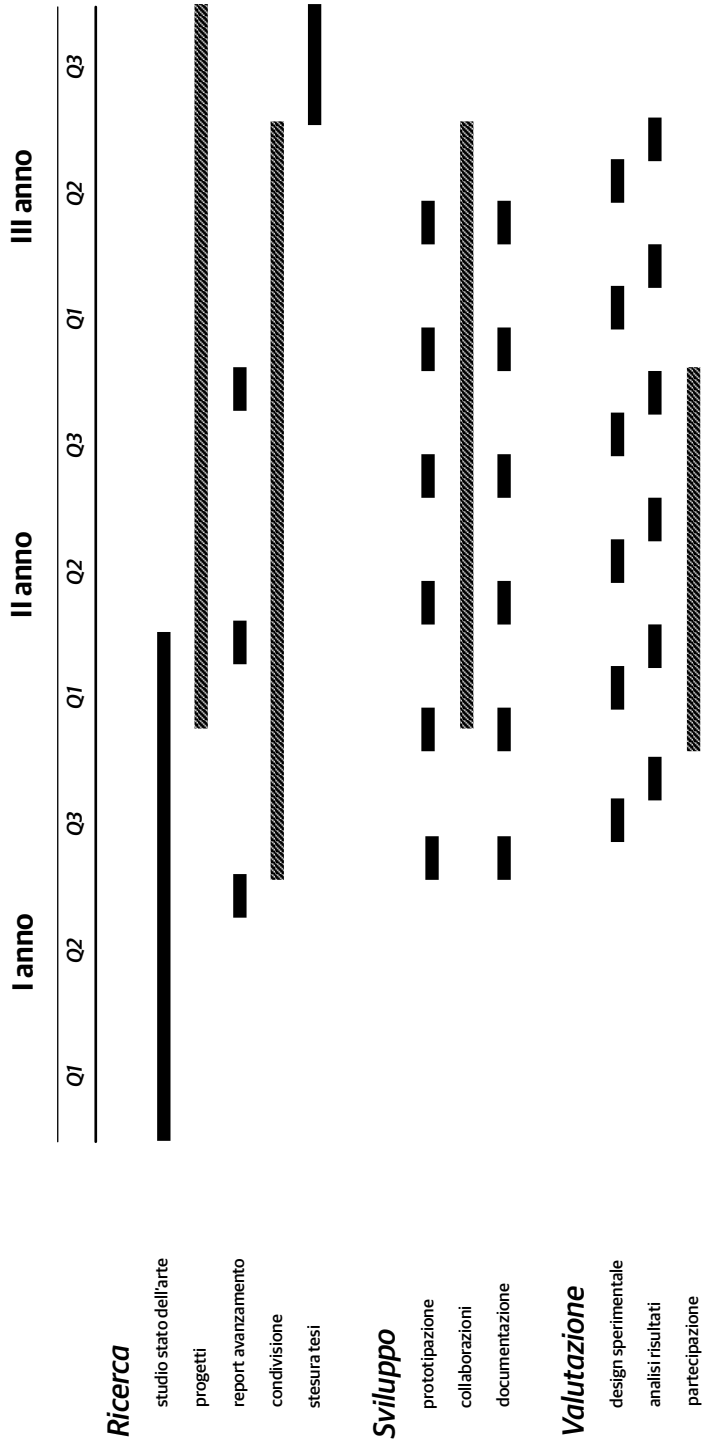

FIGURE 1. Schedulazione annuale.

esempi di training, quantità come *log-likelihood*, *pseudo-log-likelihood* and *conditional log-likelihood* verranno prese in considerazione [36, 43]. Mentre, quando si avrà a che fare con il contesto discriminativo, per esempio con un task di classificazione, si potrà fare ricorso alle misure di *accuracy*, *AUC-ROC*, *AUC-PR* [42], per un task di ranking o retrieval, invece si useranno *mean rank* o *micro/macro hits@N* [8].

Non sempre l'unico criterio per giudicare la bontà di un algoritmo sarà monitorare le sue performance nei termini finora esposti, confrontabili sono anche il tasso di convergenza dell'apprendimento (sia in termini di tempo di esecuzione che di complessità computazionale da calcolare o stimare) o il trade-off fra le performance e le risorse consumate; questo è particolarmente vero quando si parla di grandi quantità di dati e si è ben disponibili a rinunciare a qualche punto in performance per acquistarne parecchi in efficienza.

6.2. **Datasets.** Anche la scelta dei dataset da considerare in fase di design sperimentale è altamente dipendente dal task preso in questione. Per portare un esempio concreto di un setting sperimentale ben definito per un particolare task si espone qui quello di apprendimento della struttura di PGM in contesti generativi. Il confronto per questi algoritmi avviene sulla log-likelihood media computata su tutte le non viste o per rispondere a query parziali a partire da un insieme standard di dataset provenienti dai contesti di classificazione, recommending, rule mining. Esso è composto da 20 dataset ed è stato per la prima volta proposto in [28] e successivamente ampliato in [40].

References

[1] Stephen H. Bach, Bert Huang, Ben London, and Lise Getoor. Hingeloss markov random fields: Convex inference for structured prediction. In *In Uncertainty in Artificial Intelligence*, 2013.

[2] Joshua Bengio and Yan LeCun. Tutorial on deep learning, 2013.

[3] Yoshua Bengio. Learning deep architectures for ai. *Found. Trends Mach. Learn.*, 2(1):1–127, January 2009.

[4] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. Unsupervised feature learning and deep learning: A review and new perspectives. *CoRR*, abs/1206.5538, 2012.

[5] Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, Université De Montréal, and Montréal Québec. Greedy layer-wise training of deep networks. In *In NIPS*. MIT Press, 2007.

[6] Jiang Bian, Bin Gao, and Tie-Yan Liu. Knowledge-powered deep learning for word embedding. In Toon Calders, Floriana Esposito, Eyke Hüllermeier, and Rosa Meo, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 8724 of *Lecture Notes in Computer Science*, pages 132–148. Springer Berlin Heidelberg, 2014.

[7] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2006.

[8] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259, 2014.

[9] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *CoRR*, abs/1312.6203, 2013.

[10] A. Chechetka and C. Guestrin. Efficient principled learning of thin junction trees. *NIPS08*, 2008.

[11] C Chow and C Liu. Approximating discrete probability distributions with dependence trees. *Information Theory, IEEE Transactions on*, 14(3):462–467, 1968.

[12] Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun ichi Amari. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*. Wiley, 2009.

[13] Nadav Cohen and Amnon Shashua. Simnets: A generalization of convolutional networks. *CoRR*, abs/1410.0781, 2014.

[14] Adnan Darwiche. A differential approach to inference in bayesian networks. *J.ACM*, 2003.

[15] Adnan Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge, 2009.
[16] H. L. H. De Penning, A. S. D'Avila Garcez, Luís C. Lamb, and John-Jules C. Meyer. A neural-symbolic cognitive agent for online learning and reasoning. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1653–1658. AAAI Press, 2011.
[17] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE*, 41(6):391–407, 1990.
[18] Li Deng. An overview of deep-structured learning for information processing. In *Proceedings of Asian-Pacific Signal & Information Processing Annual Summit and Conference (APSIPAASC)*, 2011.
[19] Aaron Dennis and Dan Ventura. Learning the architecture of sum-product networks using clustering on varibles. *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, 2012.
[20] Pedro Domingos. A tractable first order probabilistc logic. *AAAI2012*, 2012.
[21] Xavier Frazão and LuísA. Alexandre. Dropall: Generalization of two convolutional neural network regularization methods. In Aurélio Campilho and Mohamed Kamel, editors, *Image Analysis and Recognition*, Lecture Notes in Computer Science, pages 282–289. Springer International Publishing, 2014.
[22] Nir Friedman. Learning belief networks in the presence of missing values and hidden variables. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 125–133. Morgan Kaufmann, 1997.
[23] Andrea Frome, Greg Corrado, Jon Shlens, Samy Bengio, Jeffrey Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. In *Advances In Neural Information Processing Systems, NIPS*, 2013.
[24] Lise Geetor and Ben Taskar. *Introduction to Statistical Relational Learning*. MIT Press, 2007.
[25] Robert Gens and Pedro Domingos. Discriminative learning of sum-product networks. *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, 2012.
[26] Robert Gens and Pedro Domingos. Learning the structure of sum-product networks. *International Conference on Machine Learning 30 (ICML 2013)*, 2013.
[27] Robert Gens and Pedro M. Domingos. Deep symmetry networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2537–2545, 2014.
[28] V. Gogate and Pedro Domingos. Learning efficient markov networks. *NIPS 2010*, 2010.
[29] Jan Van Haaren and Jesse Davis. Markov network structure learning: A randomized feature generation approach. In *AAAI*, 2012.
[30] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2009.
[31] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 2006.
[32] Geoffrey Hinton. Tutorial on deep belief networks. *Workshop @ NIPS2007*, 2007.
[33] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Comput.*, 18(7):1527–1554, July 2006.
[34] Charles Kemp, Noah D. Goodman, and Joshua B. Tenenbaum. Learning to learn causal models. *Cognitive Science*, 34, 2010.
[35] Charles Kemp, Joshua B. Tenenbaum, Thomas L. Griffiths, Takeshi Yamada, and Naonori Ueda. Learning systems of concepts with an infinite relational model. In *AAAI*, pages 381–388, 2006.
[36] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
[37] Yann LeCun, Sumit Chopra, Raia Hadsell, Marc'Aurelio Ranzato, and Fu-Jie Huang. A tutorial on energy-based learning. In G. Bakir, T. Hofman, B. Schölkopf, A. Smola, and B. Taskar, editors, *Predicting Structured Data*. MIT Press, 2006.
[38] Yann LeCun and Marc'Aurelio Ranzato. Deep learning tutorial. ICML, 2013. ICML.
[39] Huma Lodhi. Deep relational machines. In Minho Lee, Akira Hirose, Zeng-Guang Hou, and RheeMan Kil, editors, *Neural Information Processing*, volume 8227 of *Lecture Notes in Computer Science*, pages 212–219. Springer Berlin Heidelberg, 2013.

[40] Daniel Lowd and Jesse Davis. Learning markov network structure with decision trees. In *Proceedings of the Twenty-seventh International Conference on Machine Learning*, 2010.
[41] Daniel Lowd and Amirmohammad Rooshenas. Learning markov networks with arithmetic circuits. *JMLR W&CP 31*, 2013.
[42] Tom Mitchell. *Machine Learning*. McGraw Hill, 1997.
[43] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.
[44] Aniruddh Nath and Pedro Domingos. Learning tractable statistical relational models. *Workshop on Learning Tractable Probabilistic Graphical Models, ICML 2014*, 2014.
[45] Jiquan Ngiam, Zhenghao Chen, Pang W Koh, and Andrew Y Ng. Learning deep energy models. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1105–1112, 2011.
[46] Robert Peharz, Bernhard Geiger, and Franz Pernkopf. Greedy part-wise learning of sum-product networks. *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECMLPKDD 2013)*, 2013.
[47] Robert Peharz, Robert Gens, and Pedro Domingos. Learning selective sum-product networks. In *Workshop on Learning Tractable Probabilistic Models*. LTPM, 2014.
[48] Hoifung Poon. Tutorial on spn. *NIPS2011*, 2011.
[49] Hoifung Poon and Pedro Domingos. Sum-product network: a new deep architecture. *NIPS 2010 Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.
[50] Tahrima Rahman, Prasanna Kothalkar, and Vibhav Gogate. Cutset networks: A simple, tractable, and scalable approach for improving the accuracy of chow-liu trees. In Toon Calders, Floriana Esposito, Eyke Hüllermeier, and Rosa Meo, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 8725 of *Lecture Notes in Computer Science*, pages 630–645. Springer Berlin Heidelberg, 2014.
[51] Amirmohammad Rooshenas and Daniel Lowd. Learning tractable graphical models using mixture of arithmetic circuits. In *AAAI (Late-Breaking Developments)*, 2013.
[52] Amirmohammad Rooshenas and Daniel Lowd. Learning sum-product networks with direct and indirect variable interactions. In *Thirty-First International Conference on Machine Learning (ICML 14)*, 2014.
[53] Dan Roth and Rajhans Samdani. Learning multi-linear representations of distributions for efficient inference. *Machine Learning 75*, 2009.
[54] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In David E. Rumelhart, James L. McClelland, and CORPORATE PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1*, pages 318–362. MIT Press, Cambridge, MA, USA, 1986.
[55] Ruslan Salakhutdinov and Geoffrey Hinton. Deep boltzmann machines. *roceedings of the 12th International Confe-rence on Artificial Intelligence and Statistics (AISTATS) 2009*, 2009.
[56] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.
[57] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going Deeper with Convolutions. *ArXiv e-prints*, September 2014.
[58] Joshua B Tenenbaum, Charles Kemp, Thomas L Griffiths, and Noah D Goodman. How to grow a mind: Statistics, structure, and abstraction. *science*, 331(6022):1279–1285, 2011.
[59] Guy Van den Broeck, Nima Taghipour, Wannes Meert, Jesse Davis, and Luc De Raedt. Lifted probabilistic inference by first-order knowledge compilation. In Toby Walsh, editor, *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, International Joint Conference on Artificial Intelligence (IJCAI), Barcelona, Spain, 16-22 July 2011*, pages 2178–2185. AAAI Press/International Joint Conferences on Artificial Intelligence, July 2011.
[60] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1096–1103, New York, NY, USA, 2008. ACM.
[61] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, December 2010.
[62] Li Wan, Matthew Zeiler, Sixin Zhang, Yann L. Cun, and Rob Fergus. Regularization of neural networks using dropconnect. In Sanjoy Dasgupta and David Mcallester, editors, *Proceedings*

of the 30th International Conference on Machine Learning (ICML-13), volume 28, pages 1058–1066. JMLR Workshop and Conference Proceedings, May 2013.

[63] W. Austin Webb and Pedro Domingos. Tractable probabilistic knowledge bases with existance uncertainty. *AAAI Workshop: Statistical Relational Artificial Intelligence 2013*, 2013.

[64] Hao Wu, Martin Renqiang Min, and Bing Bai. Deep semantic embedding. In *Proceedings of Workshop on Semantic Matching in Information Retrieval co-located with the 37th international ACM SIGIR conference on research and development in information retrieval, SMIR@SIGIR 2014, Queensland, Australia, July 11, 2014.*, pages 46–52, 2014.

[65] R. Wu, S. Yan, Y. Shan, Q. Dang, and G. Sun. Deep Image: Scaling up Image Recognitionper. *ArXiv e-prints*, January 2015.

[66] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013.