

Probabilistic Reasoning via Deep Learning: Neural Association Models

Quan Liu[†] and Hui Jiang[‡] and Zhen-Hua Ling[†] and Si Wei[§] and Yu Hu^{†§}

[†] National Engineering Laboratory for Speech and Language Information Processing
University of Science and Technology of China, Hefei, Anhui, China

[‡] Department of Electrical Engineering and Computer Science, York University, Canada
[§] iFLYTEK Research, Hefei, China

emails: quanliu@mail.ustc.edu.cn, hj@cse.yorku.ca, zhling@ustc.edu.cn
siwei@iflytek.com, yuhu@iflytek.com

Abstract

In this paper, we propose a new deep learning approach, called neural association model (NAM), for probabilistic reasoning in artificial intelligence. We propose to use neural networks to model association between any two events in a domain. Neural networks take one event as input and compute a conditional probability of the other event to model how likely these two events are associated. The actual meaning of the conditional probabilities varies between applications and depends on how the models are trained. In this work, as two case studies, we have investigated two NAM structures, namely deep neural networks (DNNs) and relation-modulated neural nets (RMNNs), on several probabilistic reasoning tasks in AI, including recognizing textual entailment, triple classification in multi-relational knowledge bases and common-sense reasoning. Experimental results on several popular data sets derived from WordNet, Free-Base and ConceptNet have all demonstrated that both DNNs and RMNNs perform equally well and they can significantly outperform the conventional methods available for these reasoning tasks. Moreover, comparing with DNNs, RMNNs are superior in knowledge transfer, where a pre-trained model can be quickly extended to an unseen relation after observing only a few training samples.

1 Introduction

Reasoning is an important topic in artificial intelligence (AI), which has attracted considerable attention and research effort in the past few decades [McCarthy, 1986; Minsky, 1988; Mueller, 2014]. Besides the traditional logic reasoning, probabilistic reasoning has been studied as another typical genre in order to handle knowledge uncertainty in reasoning based on the probability theory [Pearl, 1988; Neapolitan, 2012]. The probabilistic reasoning aims to predict the conditional probability $\Pr(E_2|E_1)$ of one event E_2 given another event E_1 . State-of-the-art methods for probability reasoning include Bayesian networks [Jensen, 1996], Markov logic networks [Richardson and Domingos, 2006] and other graphical

models [Koller and Friedman, 2009]. Taking Bayesian networks as example, the conditional probabilities between two associated events are calculated as posterior probabilities according to the Bayes theorem, where all possible events are modeled by a pre-defined graph structure. However, these methods quickly become intractable for most practical tasks where the number of all possible events is large.

In recent years, distributed representations that map discrete language units into continuous vector spaces have gained significant popularity along with the development of neural networks [Bengio *et al.*, 2003; Collobert *et al.*, 2011; Mikolov *et al.*, 2013]. The main benefit for embedding in continuous space is its smooth property to capture the semantic relatedness between discrete events, potentially generalizable to unseen events. Similar ideas, such as knowledge graph embedding, have been proposed to represent knowledge bases (KB) in low-dimensional continuous space [Bordes *et al.*, 2013; Socher *et al.*, 2013; Wang *et al.*, 2014; Nickel *et al.*, 2015]. Using the smoothed KB representation, it is possible to reason over the relations among various entities. However, human-like reasoning remains as an extremely challenging problem because it requires to effectively encode world knowledge using powerful models. Most of the existing KBs are quite sparse and can only capture a fraction of world knowledge, even for those recently created large-scale KBs, such as YAGO, NELL and Freebase. In order to take advantage of these sparse knowledge bases, the state-of-the-art approaches for knowledge graph embedding usually adopt simple linear models, such as RESCAL [Nickel *et al.*, 2012] for statistical relational learning (SRL).

Motivated by the recent successes of deep learning in pattern recognition, in this paper, we propose to use deep neural networks, called *neural association model (NAM)*, to represent the sparse knowledge bases. Different from the existing linear models, the proposed NAM model uses multi-layer nonlinear activations in deep neural nets to model the association conditional probabilities between any two possible events. In the proposed NAM framework, all symbolic events are represented in low-dimensional continuous space and we have no need to explicitly specify any dependency structure among events as required in Bayesian networks. Deep neural networks are used to model the association between any two events, taking one event as input to compute a conditional probability of another event. The computed conditional

probability for association may be generalized to model various reasoning problems, such as entailment inference, relational learning, causation modelling and so on. In this work, we study two model structures for NAM. The first model is a standard deep neural networks (DNN) and the second model uses a special structure called relation modulated neural nets (RMNN). Experiments on several probabilistic reasoning tasks, including recognizing textual entailment, triple classification in multi-relational KBs and commonsense reasoning, have demonstrated that both DNN and RMNN can significantly outperform other conventional methods. Moreover, the RMNN model is shown to be effective in knowledge transfer learning, where a pre-trained model can be quickly extended to a new relation after observing only a few training samples. The main contributions of this paper can be summarized as follows:

- We propose neural association models for probabilistic reasoning based on deep neural networks, which are general enough to deal with various reasoning problems over symbolic events.
- One specific model structure studied in this work is quite effective for knowledge transfer learning, which can quickly adapt an existing knowledge base to newly encountered scenarios and contexts.
- To the best of our knowledge, this is the first work to demonstrate that DNNs with the multi-layer nonlinearity are useful for probabilistic reasoning.

2 A Review on Statistical Relation Learning

This paper mainly focuses on probabilistic reasoning on multi-relational data. We first review the formulation of statistical relational learning (SRL) as in [Nickel *et al.*, 2015]. Let $\mathcal{E} = \{e_1, \dots, e_{N_e}\}$ be the set of all entities and $\mathcal{R} = \{r_1, \dots, r_{N_r}\}$ be the set of all relation types in KB, where N_e is the total entity number and N_r the relation number. SRL attempts to model each possible triple $x_{ijk} = (e_i, r_k, e_j)$ over the sets of entities and relations as a binary random variable $y_{ijk} \in \{0, 1\}$ that indicates true or false. All possible triples in $\mathcal{E} \times \mathcal{R} \times \mathcal{E}$ may be formed as a third-order tensor $\mathbf{Y} \in \{0, 1\}^{N_e \times N_r \times N_e}$. SRL aims to estimate the joint distribution of \mathbf{Y} , $\Pr(\mathbf{Y})$, from a training set \mathcal{D} of all observed triples. Most SRL models assume that all y_{ijk} are conditionally independent given the model parameters Θ . As a result, relying on a score function $f(x_{ijk}; \Theta)$ for each triple $x_{ijk} = (e_i, r_k, e_j)$, the likelihood function of an SRL model may be derived as follows:

$$P(\mathbf{Y}|\mathcal{D}, \Theta) = \prod_{i=1}^{N_e} \prod_{j=1}^{N_e} \prod_{k=1}^{N_r} \text{Ber}(y_{ijk} | \sigma(f(x_{ijk}; \Theta))) \quad (1)$$

where $\sigma(\cdot)$ denotes the sigmoid function and $\text{Ber}(y|p)$ the Bernoulli distribution as:

$$\text{Ber}(y|p) = \begin{cases} p, & \text{if } y = 1 \\ 1 - p, & \text{if } y = 0 \end{cases} \quad (2)$$

Under the above formulation, the key work is to define the score function for each triple. In [Nickel *et al.*, 2012],

they have proposed RESCAL to explain triples via pairwise interactions of latent features. In [Sutskever *et al.*, 2009; Jenatton *et al.*, 2012], a latent factor model is designed to capture the second-order correlation between entities using a quadratic form. The work in [Lin *et al.*, 2015] attempts to learn the latent embeddings by projecting all entities from the entity space to the corresponding relation space and then building translations among the projected entities. In [Bordes *et al.*, 2012], they have proposed a model based on semantic matching energy to capture the correlation between entities and relations via multiple matrix products and Hadamard products. In [Socher *et al.*, 2013], neural tensor model (NTM) is constructed using a very expressive score function with a large number of parameters. TransE presented in [Bordes *et al.*, 2013] defines a linear function for modeling triples in a joint space of entities and relations. Finally, [Wang *et al.*, 2014] propose to construct the score function by projecting entities into a relation-specific hyperplane. Obviously, almost all existing methods in the literature adopt linear models for statistical relation learning.

3 Neural Association Models (NAM)

In this paper, we propose to use a nonlinear model, namely deep neural nets, for probabilistic reasoning. Our main goal is to use neural nets to model the association probability for any two events E_1 and E_2 in a domain, i.e., $\Pr(E_2|E_1)$ of E_2 conditioning on E_1 . All possible events in the domain are projected into continuous space without specifying any explicit dependency structure among them.

In the following, we first introduce neural association models (NAM) as a general modeling framework for probabilistic reasoning in section 3.1. Next, we describe two particular NAM model structures for multi-relational data in sections 3.2 and 3.3.

3.1 NAM in general

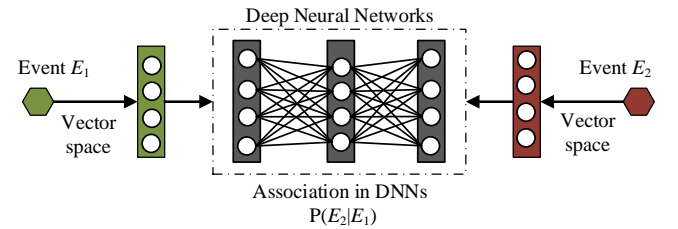


Figure 1: The NAM framework in general.

Figure 1 shows the general framework of NAM for associating two events, E_1 and E_2 . In the general NAM framework, the events are first projected into a low-dimension continuous space. Deep neural networks with multi-layer nonlinearity are used to model how likely these two events are associated. Neural networks take the embedding of one event E_1 (antecedent) as input and computes a conditional probability $\Pr(E_2|E_1)$ of the other event E_2 (consequent). If the event E_2 is binary (true or false), the NAM models may use a *sigmoid* node to compute $\Pr(E_2|E_1)$. If E_2 takes multiple

mutually exclusive values, we use a few *softmax* nodes for $\Pr(E_2|E_1)$, where it may need to use multiple embeddings for E_2 (one per value). NAMs do not explicitly specify how different events E_2 are actually related, they may be mutually exclusive, contained, intersected. NAMs are only used to separately compute conditional probabilities, $\Pr(E_2|E_1)$, for each pair of events, E_1 and E_2 , in a task. The actual physical meaning of the conditional probabilities $\Pr(E_2|E_1)$ varies between applications and depends on how the models are trained. Table 1 lists a few possible applications for NAMs.

Application	E_1	E_2
language modeling	h	w
causal reasoning	<i>cause</i>	<i>effect</i>
knowledge triple classification	$\{e_i, r_k\}$	e_j
lexical entailment	W_1	W_2
textual entailment	D_1	D_2

Table 1: Some applications for NAMs.

In language modeling, the antecedent event is the representation of history context, h , and the consequent event is next word w that takes one out of K values. In causal reasoning, E_1 and E_2 represent *cause* and *effect* respectively. For example, we have $E_1 = \text{"eating cheesy cakes"}$ and $E_2 = \text{"getting happy"}$, $\Pr(E_2|E_1)$ means that how likely E_1 may cause the binary event E_2 (true or false). In the same model, we may add more nodes to model different effects from the same E_1 , e.g., $E'_2 = \text{"growing fat"}$. Moreover, we may add 5 softmax nodes to model a multi-valued event, e.g., $E''_2 = \text{"happiness"}$ (scale from 1 to 5). Similarly, for knowledge triple classification of multi-relation data, given one triple (e_i, r_k, e_j) , E_1 consists of the head entity (*subject*) e_i and relation (*predicate*) r_k , and E_2 is a binary event indicating whether the tail entity (*object*) e_j is true or false. Finally, the applications of recognizing lexical or textual entailment, E_1 and E_2 may be defined as *premise* and *hypothesis*. More generally, NAMs can be used to model an infinite number of events E_2 , where each point in a continuous space represents a possible event. In this work, for simplicity, we only consider NAMs for a finite number of binary events E_2 but the formulation can be easily extended to more general cases.

Comparing with traditional methods, like Bayesian networks, NAMs employ neural nets as a universal approximator to directly model individual pair-wise event association probabilities without relying on explicit dependency structure among them. Therefore, NAMs can be end-to-end learned purely from training samples without strong human prior knowledge, potentially more scalable to real-world tasks.

Learning of NAMs

Assume we have a set of N_d observed examples (event pairs $\{E_1, E_2\}$), \mathcal{D} , each of which is denoted as x_n . This training set normally includes both positive and negative samples. We denote all positive samples ($E_2 = \text{true}$) as \mathcal{D}^+ and all negative samples ($E_2 = \text{false}$) as \mathcal{D}^- .

Under the same independence assumption in SRL, the log likelihood function of a NAM model can be expressed as fol-

lows:

$$\mathcal{L}(\Theta) = \sum_{x_n^+ \in \mathcal{D}^+} \ln f(x_n^+; \Theta) + \sum_{x_n^- \in \mathcal{D}^-} \ln(1 - f(x_n^-; \Theta)) \quad (3)$$

where $f(x_n; \Theta)$ denotes a *logistic* score function derived by the NAM for each x_n , which numerically computes the conditional probability $\Pr(E_2|E_1)$. More details on $f(\cdot)$ are given later. Stochastic gradient descent (SGD) methods may be used to maximize the above likelihood function, leading to maximum likelihood estimation (MLE) for NAMs.

In the following, as two case studies, we consider two NAM structures with a finite number of output nodes to model $\Pr(E_2|E_1)$ for any pair of events, where we have only a finite number of E_2 and each E_2 is binary. The first model is a typical DNN that associates antecedent event (E_1) at input and consequent event (E_2) at output. Moreover, we present another model structure, called relation-modulated neural nets, which is more suitable for multi-relational data.

3.2 DNN for NAMs

The first NAM structure is a traditional DNN as shown in Figure 2. Here we use multi-relational data in KB for illustration. Given a KB triple $x_n = (e_i, r_k, e_j)$ and its corresponding label y_n (true or false), we cast $E_1 = (e_i, r_k)$ and $E_2 = e_j$ to compute $\Pr(E_2|E_1)$ as follows.

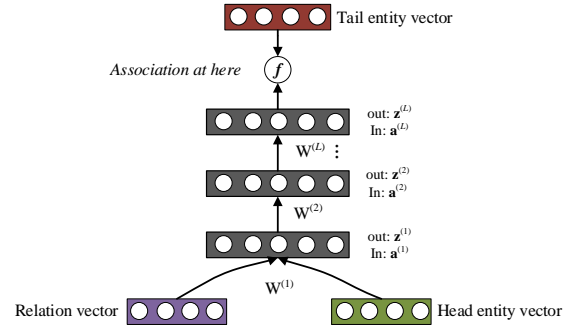


Figure 2: The DNN structure for NAMs.

Firstly, we represent head entity phrase e_i and tail entity phrase e_j by two embedding vectors $\mathbf{v}_i^{(1)} (\in \mathbf{V}^{(1)})$ and $\mathbf{v}_j^{(2)} (\in \mathbf{V}^{(2)})$. Similarly, relation r_k is also represented by a low-dimensional vector $\mathbf{c}_k \in \mathbf{C}$, which we call it *relation code* hereafter. Secondly, as shown in Figure 2, we combine the embeddings of the head entity e_i and the relation r_k to feed into an $(L + 1)$ -layer DNN as input. The DNN consists of L rectified linear (ReLU) hidden layers [Nair and Hinton, 2010]. The input to DNN is written as $\mathbf{z}^{(0)} = [\mathbf{v}_i^{(1)}, \mathbf{c}_k]$. During the feedforward process, we have

$$\mathbf{a}^{(\ell)} = \mathbf{W}^{(\ell)} \mathbf{z}^{(\ell-1)} + \mathbf{b}^{(\ell)} \quad (\ell = 1, \dots, L) \quad (4)$$

$$\mathbf{z}^{(\ell)} = h(\mathbf{a}^{(\ell)}) = \max(0, \mathbf{a}^{(\ell)}) \quad (\ell = 1, \dots, L) \quad (5)$$

where $\mathbf{W}^{(\ell)}$ and $\mathbf{b}^{(\ell)}$ represent the weight matrix and bias for layer ℓ respectively.

Finally, we propose to calculate a sigmoid score for each triple $x_n = (e_i, r_k, e_j)$ as the association probability using the last hidden layer’s output and the tail entity vector $\mathbf{v}_j^{(2)}$:

$$f(x_n; \Theta) = \sigma(\mathbf{z}^{(L)} \cdot \mathbf{v}_j^{(2)}) \quad (6)$$

where $\sigma(\cdot)$ is the *sigmoid* function, i.e., $\sigma(x) = 1/(1 + e^{-x})$.

All network parameters of this NAM structure, represented as $\Theta = \{\mathbf{W}, \mathbf{V}^{(1)}, \mathbf{V}^{(2)}, \mathbf{C}\}$, may be jointly learned by maximizing the likelihood function in eq. (3).

3.3 Relation-modulated Neural Networks (RMNN)

Particularly for multi-relation data, following the idea in [Xue *et al.*, 2014], we propose to use the so-called relation-modulated neural nets (RMNNs), as shown in Figure 3.

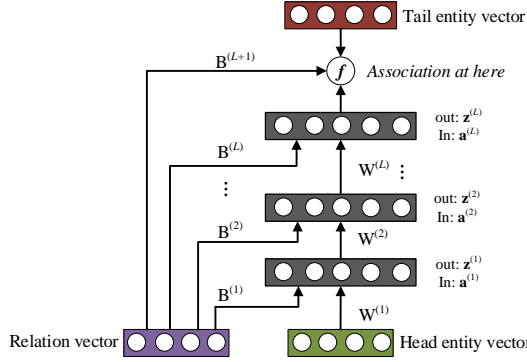


Figure 3: The relation-modulated neural networks (RMNN).

The RMNN uses the same operations as DNNs to project all entities and relations into low-dimensional continuous spaces. As shown in Figure 3, we connect the knowledge-specific relation code $\mathbf{c}^{(k)}$ to all hidden layers in the network. As shown later, this structure is superior in knowledge transfer learning tasks. Therefore, for each layer of RMNNs, instead of using eq.(4), its linear activation signal is computed from the previous layer $\mathbf{z}^{(\ell-1)}$ and the relation code $\mathbf{c}^{(k)}$ as follows:

$$\mathbf{a}^{(\ell)} = \mathbf{W}^{(\ell)} \mathbf{z}^{(\ell-1)} + \mathbf{B}^{(\ell)} \mathbf{c}^{(k)}, \quad (\ell = 1 \dots L) \quad (7)$$

where $\mathbf{W}^{(\ell)}$ and $\mathbf{B}^{(\ell)}$ represent the normal weight matrix and the relation-specific weight matrix for layer ℓ . At the topmost layer, we propose to calculate the final score for each triple $x_n = (e_i, r_k, e_j)$ using the relation code as:

$$f(x_n; \Theta) = \sigma(\mathbf{z}^{(L)} \cdot \mathbf{v}_j^{(2)} + \mathbf{B}^{(L+1)} \cdot \mathbf{c}^{(k)}). \quad (8)$$

In the same way, all RMNN parameters, including $\Theta = \{\mathbf{W}, \mathbf{B}, \mathbf{V}^{(1)}, \mathbf{V}^{(2)}, \mathbf{C}\}$, can be jointly learned based on the above maximum likelihood estimation.

The RMNN models are particularly suitable for *knowledge transfer learning*, where a pre-trained model can be quickly extended to any new relation after observing a few samples from that relation. In this case, we may estimate a new relation code based on the available new samples while keeping the whole network unchanged. Due to its small size, the

new relation code can be reliably estimated from only a small number of new samples. Furthermore, model performance in all original relations will not be affected since the model and all original relation codes are not changed in the transfer learning.

4 Experiments

In this section, we evaluate the proposed NAM models for various reasoning tasks. We first describe the experimental setup and then we will report several reasoning tasks examined in this work, including textual entailment recognition, triple classification in multi-relational KBs, commonsense reasoning and knowledge transfer learning.

4.1 Experimental setup

Here we first introduce some common experimental settings used for all experiments: (1) For entity or sentence representations, we represent them by composing from its word vectors as in [Socher *et al.*, 2013]. All word vectors are initialized from a pre-trained skip-gram [Mikolov *et al.*, 2013] word embedding model, trained on a large English Wikipedia corpus¹. The dimension for all word embeddings are set to 100 for all experiments. (2) The dimension of all relation codes are all set to 50. All relation codes are randomly initialized. (3) For network structures, we use ReLU as the non-linear activation function and all network parameters are initialized according to [Glorot and Bengio, 2010]. Meanwhile, since the number of training examples for most probabilistic reasoning tasks is relative small, we adopt the dropout approach [Hinton *et al.*, 2012] during the training process to avoid the over-fitting problem. (4) During the learning process of NAMs, we need to use negative samples, which are automatically generated by randomly perturbing positive KB triples as $\mathcal{D}^- = \{(e_i, r_k, e_\ell) | e_\ell \neq e_j \wedge (e_i, r_k, e_j) \in \mathcal{D}^+\}$.

For each task, we use the provided development set to tune for the best training hyperparameters. For example, we have tested the number of hidden layers among $\{1, 2, 3\}$, the initial learning rate among $\{0.01, 0.05, 0.1, 0.25, 0.5\}$, dropout rate among $\{0, 0.1, 0.2, 0.3, 0.4\}$. Finally, we select the best setting based on the performances on the development set: the final model structure uses 2 hidden layers, and the learning rate and the dropout rate are set to 0.1 and 0.2 for all the experiments, respectively. During model training, the learning rate is halved once the performances in the development set decreases. Both DNNs and RMNNs are trained using the stochastic gradient descend (SGD) algorithm.² We notice that the NAM models converge fast after 30 epochs of learning.

4.2 Recognizing Textual Entailment

Understanding entailment and contradiction is fundamental to understanding natural language. Here we conduct experiments on a popular recognizing textual entailment (RTE) task, which aims to recognize the entailment relationship between a pair of English sentences. In this experiment, we

¹ A small number of words do not have pre-trained vectors in the skip-gram model. We just initialize them randomly.

² All Codes for NAMs will be made available in the future.

use the SNLI dataset in [Bowman *et al.*, 2015] to conduct 2-class RTE experiments (entailment or contradiction). The SNLI dataset contains hundreds of thousands training examples, which is used to train a NAM model. Since this data set is not for multi-relational data, we only investigate the DNN structure for this task. The final NAM result, along with the baseline performance provided in [Bowman *et al.*, 2015], is listed in Table 2.

Model	Accuracy (%)
Edit Distance [Bowman <i>et al.</i> , 2015]	71.9
Classifier [Bowman <i>et al.</i> , 2015]	72.2
Lexical Resources [Bowman <i>et al.</i> , 2015]	75.0
DNN	84.7

Table 2: Experimental results on the RTE task.

From the results, we can see the proposed DNN based NAM model achieves considerable improvements over various traditional methods. This indicates that we can better model entailment relationship in natural language by representing sentences in continuous space and conducting probabilistic reasoning with deep neural networks.

4.3 Triple classification in multi-relational KBs

In this section, we evaluate the proposed NAM models on two popular knowledge triple classification datasets, namely WN11 and FB13 in [Socher *et al.*, 2013] (derived from WordNet and FreeBase), to predict whether some new triple relations hold based on other training facts in the database. The WN11 dataset contains 38,696 unique entities involving 11 different relations in total while the FB13 dataset covers 13 relations and 75,043 entities. Table 3 summarizes the statistics of these two data sets.

Dataset	# R	# Ent	# Train	# Dev	# Test
WN11	11	38,696	112,581	2,609	10,544
FB13	13	75,043	316,232	5,908	23,733

Table 3: The statistics for KBs triple classification datasets. #R is the number of relations. #Ent is the size of entity set.

The goal of knowledge triple classification is to predict whether a given triple $x_n = (e_i, r_k, e_j)$ is correct or not. We first use the training data to learn NAM models. Afterwards, we use the development set to tune a global threshold T to make a binary decision: the triple is classified as true if $f(x_n; \Theta) \geq T$; otherwise it is false. The final accuracy is calculated based on how many triplets in the test set are classified correctly. Experimental results on both WN11 and FB13 datasets are given in Table 4, where we compare the two NAM models with all other methods reported on these two data sets. The results clearly show that both NAM methods (DNN and RMNN) achieve comparable performance on these triple classification tasks, and both yield consistent improvement over all existing methods. Particularly, the RMNN model gives 3.7% and 1.9% absolute improvements over the popular neural tensor networks (NTN) [Socher *et al.*, 2013] on WN11 and FB13 respectively.

Model	WN11	FB13	Avg.
SME [Bordes <i>et al.</i> , 2012]	70.0	63.7	66.9
TransE [Bordes <i>et al.</i> , 2013]	75.9	81.5	78.7
TransH [Wang <i>et al.</i> , 2014]	78.8	83.3	81.1
TransR [Lin <i>et al.</i> , 2015]	85.9	82.5	84.2
NTN [Socher <i>et al.</i> , 2013]	86.2	90.0	88.1
DNN	89.3	91.5	90.4
RMNN	89.9	91.9	90.9

Table 4: Triple classification accuracy of various methods in WN11 and FB13.

Note that both DNN and RMNN models are much smaller than NTN in number of parameters and they scale well as the number of relation types increases. For example, either DNN or RMNN for WN11 has about 7.8 millions of parameters while NTN has about 15 millions. Although RESCAL or TransE has about 4 millions of parameters for WN11, their size goes up quickly for other tasks of thousands or more relation types. In addition, the training time of DNN and RMNN is much shorter than that of NTN or TransE since our models converge much faster. For example, we have obtained at least 5 times speedup over NTN in WN11.

4.4 Commonsense Reasoning

Similar to the triple classification task [Socher *et al.*, 2013], in this work, we use the ConceptNet KB [Liu and Singh, 2004] to construct a new commonsense data set, named as *CN14* hereafter³. When building CN14, we first select all facts in ConceptNet related to 14 typical commonsense relations, e.g., *UsedFor*, *CapableOf*. (see Figure 4 for all 14 relations.) Then, we randomly divide the extracted facts into three sets, Train, Dev and Test. Finally, in order to create a test set for classification, we randomly switch entities (in the whole vocabulary) from correct triples and result in a total of $2 \times \# \text{Test}$ triples (half is positive samples and half is negative examples). The statistics of CN14 are given in Table 5.

Dataset	# R	# Ent.	# Train	# Dev	# Test
CN14	14	159,135	200,198	5,000	10,000

Table 5: The statistics for the CN14 dataset.

The CN14 dataset is designed for answering commonsense questions like *Is a camel capable of journeying across desert?* The proposed NAM models answer this question by calculating the association probability $\Pr(E_2|E_1)$ where $E_1 = \{\text{camel, capable of}\}$ and $E_2 = \text{journey across desert}$. In this paper, we compare two NAM methods with the popular NTN method in [Socher *et al.*, 2013] on this data set and the overall results are given in Table 6. We can see that both NAM methods significantly outperform NTN in this task, and the DNN and RMNN models obtain similar performance.

Furthermore, we show the classification accuracy of all 14 relations in CN14 for RNMM and NTN in Figure 4, which

³We will publicly release this data set soon under the ConceptNet licence.

Model	Positive	Negative	total
NTN	82.7	86.5	84.6
DNN	84.5	86.9	85.7
RMNN	85.1	87.1	86.1

Table 6: Accuracy (in %) comparison on CN14.

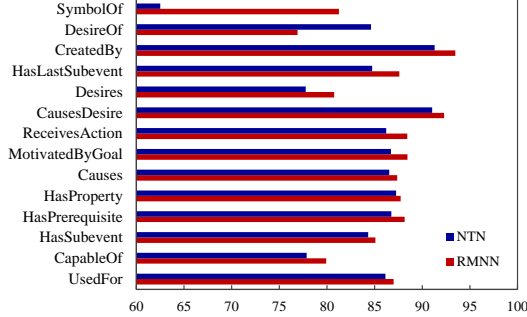


Figure 4: Accuracy of different relations in CN14.

show that the accuracy of RMNN varies among different relations from 80.1% (*Desires*) to 93.5% (*CreatedBy*). We notice some commonsense relations (such as *Desires*, *CapableOf*, *HasSubevent*) are harder than the others (like *CreatedBy*, *CausesDesire*, *MotivatedByGoal*). In general, RMNN significantly overtakes NTN in almost all relations.

4.5 Knowledge Transfer Learning

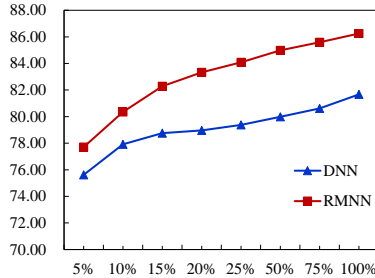


Figure 5: Accuracy (in %) on the test set of a new relation *CausesDesire* is shown as a function of used training samples from *CausesDesire* when updating the relation code only. (Accuracy on the original relations remains as 85.6%.)

Knowledge transfer between various domains is a characteristic feature and crucial cornerstone of human learning. In this section, we evaluate the proposed NAM models for a knowledge transfer learning scenario, where we adapt a pre-trained model to an unseen relation with only a few training samples from the new relation. Here we randomly select a relation, e.g., *CausesDesire* in CN14 for this experiment. This relation contains only 4800 training samples and 480 test samples. During the experiments, we use all other 13 relations in CN14 to train baseline NAM models (both DNN and RMNN). During the transfer learning, we freeze all NAM parameters, including all weights and entity representations, and only learn a new relation code for *CausesDesire* from the

given samples. At last, the learned relation code (along with the original NAM models) is used to classify the new samples of *CausesDesire* in the test set. Obviously, this transfer learning does not affect the model performance in the original 13 relations because the models are not changed. Figure 5 show the results of knowledge transfer learning for the relation *CausesDesire* as we increase the training samples gradually. The result shows that RMNN performs much better than DNN in this experiment, where we can significantly improve RMNN for the new relation with only 5-20% of the total training samples for *CausesDesire*. This demonstrates that the structure to connect the relation code to all hidden layers leads to more effective learning of new relation codes from a relatively small number of training samples.

Next, we also experiment a more aggressive learning strategy for this transfer learning setting, where we simultaneously update all the network parameters during the learning of the new relation code. The results are shown in Figure 6. This strategy can definitely improve performance more on the new relation, especially when we add more training samples. However, as expected, the performance on the original 13 relations will deteriorate. The DNN may improve the performance on the new relation as we use all training samples (up to 94.6%). However, the performance on the remain 13 original relations drops dramatically from 85.6% to 75.5%. Once again, RMNN shows the advantage over DNN in this transfer learning setting, where the accuracy on the new relation increases from 77.9% to 90.8% but the accuracy on the original 13 relations only drop slightly from 85.9% to 82.0%.

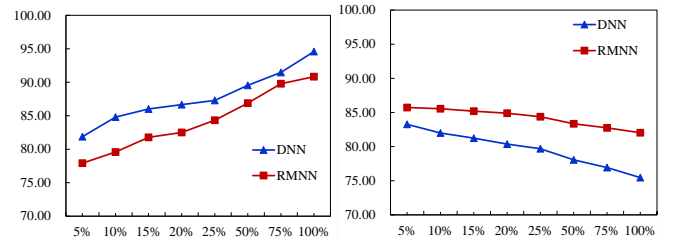


Figure 6: Transfer learning results by updating all network parameters. The left figure shows results on the new relation while the right figure shows results on the original relations.

5 Conclusions

In this paper, we have proposed neural association models (NAM) for probabilistic reasoning. We use neural networks to model association probabilities between any two events in a domain. In this work, we have investigated two model structures, namely DNN and RMNN, for NAMs. Experimental results on several reasoning tasks have shown that both DNNs and RMNNs can significantly outperform the existing methods. This paper also reports some preliminary results to use NAMs for knowledge transfer learning. We have found that the proposed RMNN model can be quickly adapted to a new relation without sacrificing the performance in the original relations.

References

- [Bengio *et al.*, 2003] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.
- [Bordes *et al.*, 2012] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. Joint learning of words and meaning representations for open-text semantic parsing. In *Proceedings of AISTATS*, pages 127–135, 2012.
- [Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of NIPS*, pages 2787–2795, 2013.
- [Bowman *et al.*, 2015] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.
- [Collobert *et al.*, 2011] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.
- [Glorot and Bengio, 2010] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of AISTATS*, pages 249–256, 2010.
- [Hinton *et al.*, 2012] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.
- [Jenatton *et al.*, 2012] Rodolphe Jenatton, Nicolas L Roux, Antoine Bordes, and Guillaume R Obozinski. A latent factor model for highly multi-relational data. In *Proceedings of NIPS*, pages 3167–3175, 2012.
- [Jensen, 1996] Finn V Jensen. *An introduction to Bayesian networks*, volume 210. UCL press London, 1996.
- [Koller and Friedman, 2009] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.
- [Lin *et al.*, 2015] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of AAAI*, 2015.
- [Liu and Singh, 2004] Hugo Liu and Push Singh. Conceptnet: a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004.
- [McCarthy, 1986] John McCarthy. Applications of circumscription to formalizing common-sense knowledge. *Artificial Intelligence*, 28(1):89–116, 1986.
- [Mikolov *et al.*, 2013] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
- [Minsky, 1988] Marvin Minsky. *Society of mind*. Simon and Schuster, 1988.
- [Mueller, 2014] Erik T Mueller. *Commonsense Reasoning: An Event Calculus Based Approach*. Morgan Kaufmann, 2014.
- [Nair and Hinton, 2010] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of ICML*, pages 807–814, 2010.
- [Neapolitan, 2012] Richard E Neapolitan. *Probabilistic reasoning in expert systems: theory and algorithms*. CreateSpace Independent Publishing Platform, 2012.
- [Nickel *et al.*, 2012] Maximilian Nickel, Volker Tresp, and Hans-Peter Krieger. Factorizing YAGO: scalable machine learning for linked data. In *Proceedings of WWW*, pages 271–280. ACM, 2012.
- [Nickel *et al.*, 2015] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *arXiv preprint arXiv:1503.00759*, 2015.
- [Pearl, 1988] Judea Pearl. Probabilistic reasoning in intelligent systems: Networks of plausible reasoning, 1988.
- [Richardson and Domingos, 2006] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine learning*, 62(1-2):107–136, 2006.
- [Socher *et al.*, 2013] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Proceedings of NIPS*, pages 926–934, 2013.
- [Sutskever *et al.*, 2009] Ilya Sutskever, Joshua B Tenenbaum, and Ruslan R Salakhutdinov. Modelling relational data using bayesian clustered tensor factorization. In *Proceedings of NIPS*, pages 1821–1828, 2009.
- [Wang *et al.*, 2014] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of AAAI*, pages 1112–1119. Citeseer, 2014.
- [Xue *et al.*, 2014] Shaofei Xue, Ossama Abdel-Hamid, Hui Jiang, Lirong Dai, and Qingfeng Liu. Fast adaptation of deep neural network based on discriminant codes for speech recognition. *Audio, Speech, and Language Processing, IEEE/ACM Trans. on*, 22(12):1713–1725, 2014.