Learning New Semi-Supervised Deep Auto-encoder Features for Statistical Machine Translation

Shixiang Lu, Zhenbiao Chen, Bo Xu
Interactive Digital Media Technology Research Center (IDMTech)
Institute of Automation, Chinese Academy of Sciences, Beijing, China
{shixiang.lu,zhenbiao.chen,xubo}@ia.ac.cn

Abstract

In this paper, instead of designing new features based on intuition, linguistic knowledge and domain, we learn some new and effective features using the deep auto-encoder (DAE) paradigm for phrase-based translation model. Using the unsupervised pre-trained deep belief net (DBN) to initialize DAE's parameters and using the input original phrase features as a teacher for semi-supervised fine-tuning, we learn new semi-supervised DAE features, which are more effective and stable than the unsupervised DBN features. Moreover, to learn high dimensional feature representation, we introduce a natural horizontal composition of more DAEs for large hidden layers feature learning. On two Chinese-English tasks, our semi-supervised DAE features obtain statistically significant improvements of 1.34/2.45 (IWSLT) and 0.82/1.52 (NIST) BLEU points over the unsupervised DBN features and the baseline features, respectively.

1 Introduction

Recently, many new features have been explored for SMT and significant performance have been obtained in terms of translation quality, such as syntactic features, sparse features, and reordering features. However, most of these features are manually designed on linguistic phenomena that are related to bilingual language pairs, thus they are very difficult to devise and estimate.

Instead of designing new features based on intuition, linguistic knowledge and domain, for the first time, Maskey and Zhou (2012) explored the possibility of inducing new features in an unsupervised fashion using deep belief net (DBN) (Hinton et al., 2006) for hierarchical phrase-based translation model. Using the 4 original phrase features in the phrase table as the input features, they pre-trained the DBN by contrastive divergence (Hinton, 2002), and generated new unsupervised DBN features using forward computation. These new features are appended as extra features to the phrase table for the translation decoder.

However, the above approach has two major shortcomings. First, the input original features for the DBN feature learning are too simple, the limited 4 phrase features of each phrase pair, such as bidirectional phrase translation probability and bidirectional lexical weighting (Koehn et al., 2003), which are a bottleneck for learning effective feature representation. Second, it only uses the unsupervised layer-wise pre-training of DBN built with stacked sets of Restricted Boltzmann Machines (RBM) (Hinton, 2002), does not have a training objective, so its performance relies on the empirical parameters. Thus, this approach is unstable and the improvement is limited. In this paper, we strive to effectively address the above two shortcomings, and systematically explore the possibility of learning new features using deep (multilayer) neural networks (DNN, which is usually referred under the name *Deep Learning*) for SMT.

To address the first shortcoming, we adapt and extend some simple but effective phrase features as the input features for new DNN feature learning, and these features have been shown significant improvement for SMT, such as, phrase pair similarity (Zhao et al., 2004), phrase frequency, phrase length (Hopkins and May, 2011), and phrase generative probability (Foster et al., 2010), which also show further improvement for new phrase feature learning in our experiments.

To address the second shortcoming, inspired by the successful use of DAEs for handwritten digits recognition (Hinton and Salakhutdinov, 2006; Hinton et al., 2006), information retrieval (Salakhutdinov and Hinton, 2009; Mirowski et

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, pages 122–132,
Baltimore, Maryland, USA, June 23-25 2014.

al., 2010), and speech spectrograms (Deng et al., 2010), we propose new feature learning using semi-supervised DAE for phrase-based translation model. By using the input data as the teacher, the "semi-supervised" fine-tuning process of DAE addresses the problem of "back-propagation without a teacher" (Rumelhart et al., 1986), which makes the DAE learn more powerful and abstract features (Hinton and Salakhutdinov, 2006). For our semi-supervised DAE feature learning task, we use the unsupervised pre-trained DBN to initialize DAE's parameters and use the input original phrase features as the "teacher" for semi-supervised back-propagation. Compared with the unsupervised DBN features, our semi-supervised DAE features are more effective and stable.

Moreover, to learn high dimensional feature representation, we introduce a natural horizontal composition for DAEs (HCDAE) that can be used to create large hidden layer representations simply by horizontally combining two (or more) DAEs (Baldi, 2012), which shows further improvement compared with single DAE in our experiments.

It is encouraging that, non-parametric feature expansion using gaussian mixture model (GMM) (Nguyen et al., 2007), which guarantees invariance to the specific embodiment of the original features, has been proved as a feasible feature generation approach for SMT. Deep models such as DNN have the potential to be much more representationally efficient for feature learning than shallow models like GMM. Thus, instead of GMM, we use DNN (DBN, DAE and HCDAE) to learn new non-parametric features, which has the similar evolution in speech recognition (Dahl et al., 2012; Hinton et al., 2012). DNN features are learned from the non-linear combination of the input original features, they strong capture high-order correlations between the activities of the original features, and we believe this deep learning paradigm induces the original features to further reach their potential for SMT.

Finally, we conduct large-scale experiments on IWSLT and NIST Chinese-English translation tasks, respectively, and the results demonstrate that our solutions solve the two aforementioned shortcomings successfully. Our semi-supervised DAE features significantly outperform the unsupervised DBN features and the baseline features, and our introduced input phrase features significantly improve the performance of DAE feature learning.

The remainder of this paper is organized as follows. Section 2 briefly summarizes the recent related work about the applications of DNN for SMT tasks. Section 3 presents our introduced input features for DNN feature learning. Section 4 describes how to learn our semi-supervised DAE features for SMT. Section 5 describes and discusses the large-scale experimental results. Finally, we end with conclusions in section 6.

2 Related Work

Recently, there has been growing interest in use of DNN for SMT tasks. Le et al. (2012) improved translation quality of n-gram translation model by using a bilingual neural LM, where translation probabilities are estimated using a continuous representation of translation units in lieu of standard discrete representations. Kalchbrenner and Blunsom (2013) introduced recurrent continuous translation models that comprise a class for purely continuous sentence-level translation models. Auli et al. (2013) presented a joint language and translation model based on a recurrent neural network which predicts target words based on an unbounded history of both source and target words. Liu et al. (2013) went beyond the log-linear model for SMT and proposed a novel additive neural networks based translation model, which overcome some of the shortcomings suffered by the log-linear model: linearity and the lack of deep interpretation and representation in features. Li et al. (2013) presented an ITG reordering classifier based on recursive autoencoders, and generated vector space representations for variable-sized phrases, which enable predicting orders to exploit syntactic and semantic information. Lu et al. (2014) adapted and extended the max-margin based RNN (Socher et al., 2011) into HPB translation with force decoding and converting tree, and proposed a RNN based word topology model for HPB translation, which successfully capture the topological structure of the words on the source side in a syntactically and semantically meaningful order.

However, none of these above works have focused on learning new features automatically with input data, and while learning suitable features (representations) is the superiority of DNN since it has been proposed. In this paper, we systematically explore the possibility of learning new fea-

tures using DNN for SMT.

3 Input Features for DNN Feature Learning

The phrase-based translation model (Koehn et al., 2003; Och and Ney, 2004) has demonstrated superior performance and been widely used in current SMT systems, and we employ our implementation on this translation model. Next, we adapt and extend some original phrase features as the input features for DAE feature learning.

3.1 Baseline phrase features

We assume that source phrase $f = f_1, \cdots, f_{l_f}$ and target phrase $e = e_1, \cdots, e_{l_e}$ include l_f and l_e words, respectively. Following (Maskey and Zhou, 2012), we use the following 4 phrase features of each phrase pair (Koehn et al., 2003) in the phrase table as the first type of input features, bidirectional phrase translation probability ($P(e|f)$ and $P(f|e)$), bidirectional lexical weighting ($Lex(e|f)$ and $Lex(f|e)$),

$$X_1 \rightarrow P(f|e), Lex(f|e), P(e|f), Lex(e|f)$$

3.2 Phrase pair similarity

Zhao et al. (2004) proposed a way of using term weight based models in a vector space as additional evidences for phrase pair translation quality. This model employ phrase pair similarity to encode the weights of content and non-content words in phrase translation pairs. Following (Zhao et al., 2004), we calculate bidirectional phrase pair similarity using cosine distance and BM25 distance as,

$$S_i^{cos}(e,f) = \frac{\sum_{j=1}^{l_e}\sum_{i=1}^{l_f} w_{e_j} p(e_j|f_i) w_{f_i}}{sqrt(\sum_{j=1}^{l_e} w_{e_j}^2) sqrt(\sum_{j=1}^{l_e} {w_a^{e_j}}^2)}$$

$$S_d^{cos}(f,e) = \frac{\sum_{i=1}^{l_f}\sum_{j=1}^{l_e} w_{f_i} p(f_i|e_j) w_{e_j}}{sqrt(\sum_{i=1}^{l_f} w_{f_i}^2) sqrt(\sum_{i=1}^{l_f} {w_a^{f_i}}^2)}$$

where, $p(e_j|f_i)$ and $p(f_i|e_j)$ represents bidirectional word translation probability. w_{f_i} and w_{e_j} are term weights for source and target words, $w_a^{e_j}$ and $w_a^{f_i}$ are the transformed weights mapped from all source/target words to the target/source dimension at word e_j and f_i, respectively.

$$S_i^{bm25}(e,f) = \sum_{i=1}^{l_f} idf_{f_i} \frac{(k_1+1) w_{f_i} (k_3+1) w_a^{f_i}}{(K + w_{f_i})(k_3 + w_a^{f_i})}$$

$$S_d^{bm25}(f,e) = \sum_{j=1}^{l_e} idf_{e_j} \frac{(k_1+1) w_{e_j} (k_3+1) w_a^{e_j}}{(K + w_{e_j})(k_3 + w_a^{e_j})}$$

where, k_1, b, k_3 are set to be 1, 1 and 1000, respectively. $K = k_1((1-b) + J/avg(l))$, and J is the phrase length (l_e or l_f), $avg(l)$ is the average phrase length. Thus, we have the second type of input features

$$X_2 \rightarrow S_i^{cos}(f,e), S_i^{bm25}(f,e), S_d^{cos}(e,f), S_d^{bm25}(e,f)$$

3.3 Phrase generative probability

We adapt and extend bidirectional phrase generative probabilities as the input features, which have been used for domain adaptation (Foster et al., 2010). According to the background LMs, we estimate the bidirectional (source/target side) forward and backward phrase generative probabilities as

$$P_f(f) = P(f_1)P(f_2|f_1)\cdots P(f_{l_f}|f_{l_f-n+1}, \cdots, f_{l_f-1})$$
$$P_f(e) = P(e_1)P(e_2|e_1)\cdots P(e_{l_e}|e_{l_e-n+1}, \cdots, e_{l_e-1})$$
$$P_b(f) = P(f_{l_f})P(f_{l_f-1}|f_{l_f})\cdots P(f_1|f_n, \cdots, f_2)$$
$$P_b(e) = P(e_{l_e})P(e_{l_e-1}|e_{l_e})\cdots P(e_1|e_n, \cdots, e_2)$$

where, the bidirectional forward and backward[1] background 4-gram LMs are trained by the corresponding side of bilingual corpus[2]. Then, we have the third type of input features

$$X_3 \rightarrow P_f(e), P_b(e), P_f(f), P_b(f)$$

3.4 Phrase frequency

We consider bidirectional phrase frequency as the input features, and estimate them as

$$P(f) = \frac{count(f)}{\sum_{|f_i|=|f|} count(f_i)}$$

$$P(e) = \frac{count(e)}{\sum_{|e_j|=|e|} count(e_j)}$$

where, the $count(f)$/$count(e)$ are the total number of phrase f/e appearing in the source/target side of the bilingual corpus, and the denominator are the total number of the phrases whose length are equal to $|f|$/$|e|$, respectively. Then, we have the forth type of input features

$$X_4 \rightarrow P(f), P(e)$$

[1] Backward LM has been introduced by Xiong et al. (2011), which successfully capture both the preceding and succeeding contexts of the current word, and we estimate the backward LM by inverting the order in each sentence in the training data from the original order to the reverse order.

[2] This corpus is used to train the translation model in our experiments, and we will describe it in detail in section 5.1.

3.5 Phrase length

Phrase length plays an important role in the translation process (Koehn, 2010; Hopkins and May, 2011). We normalize bidirectional phrase length by the maximum phrase length, and introduce them as the last type of input features

$$X_5 \rightarrow l_e^n, l_f^n$$

In summary, except for the first type of phrase feature X_1 which is used by (Maskey and Zhou, 2012), we introduce another four types of effective phrase features X_2, X_3, X_4 and X_5. Now, the input original phrase features X includes 16 features in our experiments, as follows,

$$X \rightarrow X_1, X_2, X_3, X_4, X_5$$

We build the DAE network where the first layer with visible nodes equaling to 16, and each visible node v_i corresponds to the above original features X in each phrase pair.

4 Semi-Supervised Deep Auto-encoder Features Learning for SMT

Each translation rule in the phrase-based translation model has a set number of features that are combined in the log-linear model (Och and Ney, 2002), and our semi-supervised DAE features can also be combined in this model. In this section, we design our DAE network with various network structures for new feature learning.

4.1 Learning a Deep Belief Net

Inspired by (Maskey and Zhou, 2012), we first learn a deep generative model for feature learning using DBN. DBN is composed of multiple layers of latent variables with the first layer representing the visible feature vectors, which is built with stacked sets of RBMs (Hinton, 2002).

For a RBM, there is full connectivity between layers, but no connections within either layer. The connection weight W, hidden layer biases c and visible layer biases b can be learned efficiently using the contrastive divergence (Hinton, 2002; Carreira-Perpinan and Hinton, 2005). When given a hidden layer h, factorial conditional distribution of visible layer v can be estimated by

$$P(v = 1|h) = \sigma(b + h^T W^T)$$

where σ denotes the logistic sigmoid. Given v, the element-wise conditional distribution of h is

$$P(h = 1|v) = \sigma(c + v^T W)$$

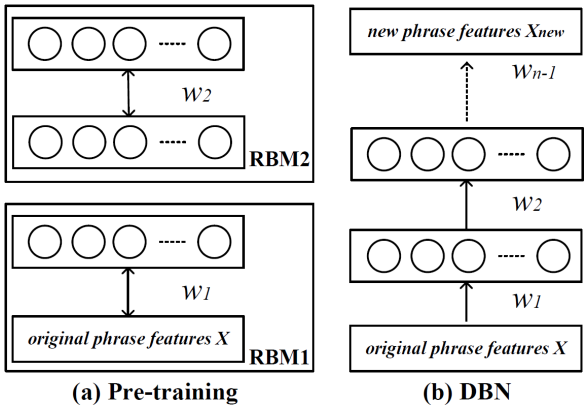


Figure 1: Pre-training consists of learning a stack of RBMs, and these RBMs create an unsupervised DBN.

The two conditional distributions can be shown to correspond to the generative model,

$$P(v, h) = \frac{1}{Z} exp(-E(v, h))$$

where,

$$Z = \sum_{v,h} e^{-E(v,h)}$$

$$E(v, h) = -b^T v - c^T h - v^T W h$$

After learning the first RBM, we treat the activation probabilities of its hidden units, when they are being driven by data, as the data for training a second RBM. Similarly, a n_{th} RBM is built on the output of the $n - 1_{th}$ one and so on until a sufficiently deep architecture is created. These n RBMs can then be composed to form a DBN in which it is easy to infer the states of the n_{th} layer of hidden units from the input in a single forward pass (Hinton et al., 2006), as shown in Figure 1. This greedy, layer-by-layer pre-training can be repeated several times to learn a deep, hierarchical model (DBN) in which each layer of features captures strong high-order correlations between the activities of features in the layer below.

To deal with real-valued input features X in our task, we use an RBM with Gaussian visible units (GRBM) (Dahl et al., 2012) with a variance of 1 on each dimension. Hence, $P(v|h)$ and $E(v, h)$ in the first RBM of DBN need to be modified as

$$P(v|h) = \mathcal{N}(v; b + h^T W^T, I)$$

$$E(v, h) = \frac{1}{2}(v - b)^T (v - b) - c^T h - v^T W h$$

where I is the appropriate identity matrix.

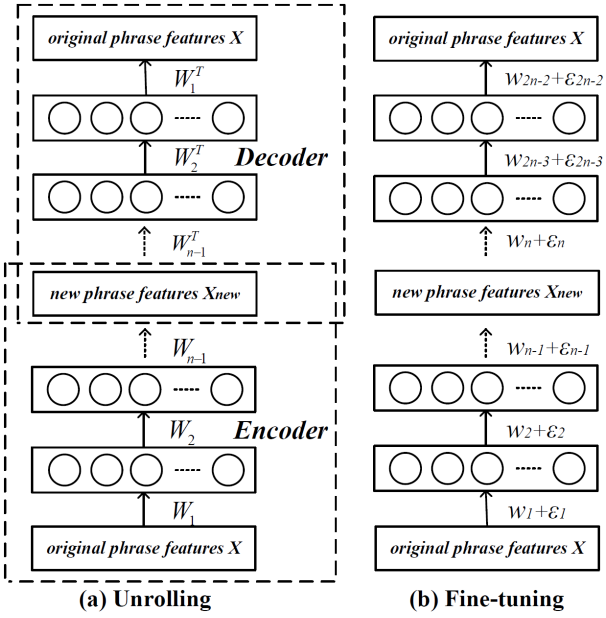


Figure 2: After the unsupervised pre-training, the DBNs are "unrolled" to create a semi-supervised DAE, which is then fine-tuned using back-propagation of error derivatives.

To speed-up the pre-training, we subdivide the entire phrase pairs (with features X) in the phrase table into small mini-batches, each containing 100 cases, and update the weights after each mini-batch. Each layer is greedily pre-trained for 50 epochs through the entire phrase pairs. The weights are updated using a learning rate of 0.1, momentum of 0.9, and a weight decay of 0.0002 $\times$ weight $\times$ learning rate. The weight matrix W are initialized with small random values sampled from a zero-mean normal distribution with variance 0.01.

After the pre-training, for each phrase pair in the phrase table, we generate the DBN features (Maskey and Zhou, 2012) by passing the original phrase features X through the DBN using forward computation.

4.2 From DBN to Deep Auto-encoder

To learn a semi-supervised DAE, we first "unroll" the above n layer DBN by using its weight matrices to create a deep, 2n-1 layer network whose lower layers use the matrices to "encode" the input and whose upper layers use the matrices in reverse order to "decode" the input (Hinton and Salakhutdinov, 2006; Salakhutdinov and Hinton, 2009; Deng et al., 2010), as shown in Figure 2. The layer-wise learning of DBN as above must be treated as a pre-training stage that finds a good region of the parameter space, which is used to initialize our DAE's parameters. Starting in this region, the DAE is then fine-tuned using average squared error (between the output and input) back-propagation to minimize reconstruction error, as to make its output as equal as possible to its input.

For the fine-tuning of DAE, we use the method of conjugate gradients on larger mini-batches of 1000 cases, with three line searches performed for each mini-batch in each epoch. To determine an adequate number of epochs and to avoid over-fitting, we fine-tune on a fraction phrase table and test performance on the remaining validation phrase table, and then repeat fine-tuning on the entire phrase table for 100 epochs.

We experiment with various values for the noise variance and the threshold, as well as the learning rate, momentum, and weight-decay parameters used in the pre-training, the batch size and epochs in the fine-tuning. Our results are fairly robust to variations in these parameters. The precise weights found by the pre-training do not matter as long as it finds a good region of the parameter space from which to start the fine-tuning.

The fine-tuning makes the feature representation in the central layer of the DAE work much better (Salakhutdinov and Hinton, 2009). After the fine-tuning, for each phrase pair in the phrase table, we estimate our DAE features by passing the original phrase features X through the "encoder" part of the DAE using forward computation.

To combine these learned features (DBN and DAE feature) into the log-linear model, we need to eliminate the impact of the non-linear learning mechanism. Following (Maskey and Zhou, 2012), these learned features are normalized by the average of each dimensional respective feature set. Then, we append these features for each phrase pair to the phrase table as extra features.

4.3 Horizontal Composition of Deep Auto-encoders (HCDAE)

Although DAE can learn more powerful and abstract feature representation, the learned features usually have smaller dimensionality compared with the dimensionality of the input features, such as the successful use for handwritten digits recognition (Hinton and Salakhutdinov, 2006; Hinton et al., 2006), information retrieval (Salakhutdinov and Hinton, 2009; Mirowski et al., 2010), and

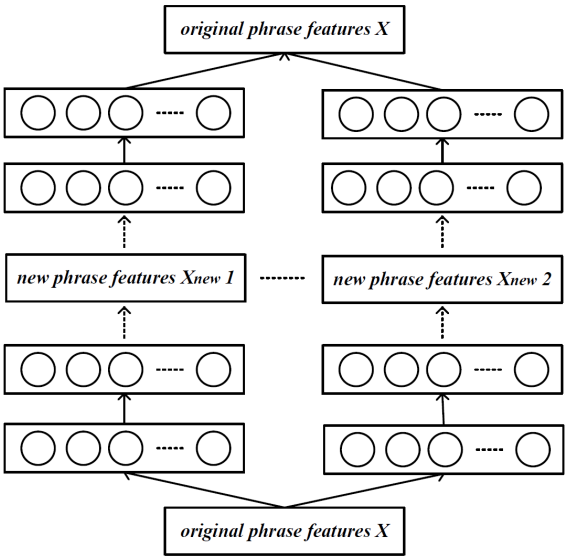


Figure 3: Horizontal composition of DAEs to expand high-dimensional features learning.

4-16-8-2	4-16-8-4	4-16-16-8
4-16-8-4-2	4-16-16-8-4	4-16-16-8-8
4-16-16-8-4-2	4-16-16-8-8-4	4-16-16-16-8-8
4-16-16-8-8-4-2	4-16-16-16-8-8-4	4-16-16-16-16-8-8
6-16-8-2	6-16-8-4	6-16-16-8
6-16-8-4-2	6-16-16-8-4	6-16-16-8-8
6-16-16-8-4-2	6-16-16-8-8-4	6-16-16-16-8-8
6-16-16-16-8-4-2	6-16-16-16-8-8-4	6-16-16-16-16-8-8
8-16-8-2	8-16-8-4	8-16-16-8
8-16-8-4-2	8-16-16-8-4	8-16-16-8-8
8-16-16-8-4-2	8-16-16-8-8-4	8-16-16-16-8-8
8-16-16-16-8-4-2	8-16-16-16-8-8-4	8-16-16-16-16-8-8
16-32-16-2	16-32-16-4	16-32-16-8
16-32-16-8-2	16-32-16-8-4	16-32-32-16-8
16-32-16-8-4-2	16-32-32-16-8-4	16-32-32-16-16-8
16-32-32-16-8-4-2	16-32-32-16-16-8-4	16-32-32-32-16-16-8

Table 1: Details of the used network structure. For example, the architecture 16-32-16-2 (4 layers' network depth) corresponds to the DAE with 16-dimensional input features (X) (input layer), 32/16 hidden units (first/second hidden layer), and 2-dimensional output features (new DAE features) (output layer). During the fine-tuning, the DAE's network structure becomes 16-32-16-2-16-32-16. Correspondingly, 4-16-8-2 and 6(8)-16-8-2 represent the input features are X_1 and X_1+X_i.

speech spectrograms (Deng et al., 2010). Moreover, although we have introduced another four types of phrase features (X_2, X_3, X_4 and X_5), the only 16 features in X are a bottleneck for learning large hidden layers feature representation, because it has limited information, the performance of the high-dimensional DAE features which are directly learned from single DAE is not very satisfactory.

To learn high-dimensional feature representation and to further improve the performance, we introduce a natural horizontal composition for DAEs that can be used to create large hidden layer representations simply by horizontally combining two (or more) DAEs (Baldi, 2012), as shown in Figure 3. Two single DAEs with architectures $16/m_1/16$ and $16/m_2/16$ can be trained and the hidden layers can be combined to yield an expanded hidden feature representation of size $m_1 + m_2$, which can then be fed to the subsequent layers of the overall architecture. Thus, these new $m_1 + m_2$-dimensional DAE features are added as extra features to the phrase table.

Differences in m_1- and m_2-dimensional hidden representations could be introduced by many different mechanisms (e.g., learning algorithms, initializations, training samples, learning rates, or distortion measures) (Baldi, 2012). In our task, we introduce differences by using different initializations and different fractions of the phrase table.

5 Experiments and Results

5.1 Experimental Setup

We now test our DAE features on the following two Chinese-English translation tasks.

IWSLT. The bilingual corpus is the Chinese-English part of Basic Traveling Expression corpus (BTEC) and China-Japan-Korea (CJK) corpus (0.38M sentence pairs with 3.5/3.8M Chinese/English words). The LM corpus is the English side of the parallel data (BTEC, CJK and CWMT08[3]) (1.34M sentences). Our development set is IWSLT 2005 test set (506 sentences), and our test set is IWSLT 2007 test set (489 sentences).

NIST. The bilingual corpus is LDC[4] (3.4M sentence pairs with 64/70M Chinese/English words). The LM corpus is the English side of the parallel data as well as the English Gigaword corpus (LDC2007T07) (11.3M sentences). Our development set is NIST 2005 MT evaluation set (1084 sentences), and our test set is NIST 2006 MT evaluation set (1664 sentences).

We choose the Moses (Koehn et al., 2007) framework to implement our phrase-based machine system. The 4-gram LMs are estimated by the SRILM toolkit with modified Kneser-Ney

[3]the 4th China Workshop on Machine Translation

[4]LDC2002E18, LDC2002T01, LDC2003E07, LDC2003E14, LDC2003T17, LDC2004T07, LDC2004T08, LDC2005T06, LDC2005T10, LDC2005T34, LDC2006T04, LDC2007T09

#		Features	IWSLT		NIST	
			Dev	Test	Dev	Test
1		Baseline	50.81	41.13	36.12	32.59
2	X_1	+DBN_X_1_2f	51.92	42.07*	36.33	33.11*
3		+DAE_X_1_2f	52.49	43.22**	36.92	33.44**
4		+DBN_X_1_4f	51.45	41.78*	36.45	33.12*
5		+DAE_X_1_4f	52.45	43.06**	36.88	33.47**
6		+HCDAE_X_1_2+2f	53.69	43.23***	37.06	33.68***
7		+DBN_X_1_8f	51.74	41.85*	36.61	33.24*
8		+DAE_X_1_8f	52.33	42.98**	36.81	33.36**
9		+HCDAE_X_1_4+4f	52.52	43.26***	37.01	33.63***
10	X	+DBN_X_2f	52.21	42.24*	36.72	33.21*
11		+DAE_X_2f	52.86	43.45**	37.39	33.83**
12		+DBN_X_4f	51.83	42.08*	34.45	33.07*
13		+DAE_X_4f	52.81	43.47**	37.48	33.92**
14		+HCDAE_X_2+2f	53.05	43.58***	37.59	34.11***
15		+DBN_X_8f	51.93	42.01*	36.74	33.29*
16		+DAE_X_8f	52.69	43.26**	37.36	33.75**
17		+HCDAE_X_4+4f	52.93	43.49***	37.53	34.02***
18		+(X_2+X_3+X_4+X_5)	52.23	42.91*	36.96	33.65*
19		+(X_2+X_3+X_4+X_5)+DAE_X_2f	53.55	44.17^{+***}	38.23	34.50^{+***}
20		+(X_2+X_3+X_4+X_5)+DAE_X_4f	53.61	44.22^{+***}	38.28	34.47^{+***}
21		+(X_2+X_3+X_4+X_5)+HCDAE_X_2+2f	53.75	44.28^{+****}	38.35	34.65^{+****}
22		+(X_2+X_3+X_4+X_5)+DAE_X_8f	53.47	44.19^{+***}	38.26	34.46^{+***}
23		+(X_2+X_3+X_4+X_5)+HCDAE_X_4+4f	53.62	44.29^{+****}	38.39	34.57^{+****}

Table 2: The translation results by adding new DNN features (DBN feature (Maskey and Zhou, 2012), our proposed DAE and HCDAE feature) as extra features to the phrase table on two tasks. "DBN_X_1_xf", "DBN_X_xf", "DAE_X_1_xf" and "DAE_X_xf" represent that we use DBN and DAE, input features X_1 and X, to learn x-dimensional features, respectively. "HCDAE_X_x+xf" represents horizontally combining two DAEs and each DAE has the same x-dimensional learned features. All improvements on two test sets are statistically significant by the bootstrap resampling (Koehn, 2004). *: significantly better than the baseline ($p < 0.05$), **: significantly better than "DBN_X_1_xf" or "DBN_X_xf" ($p < 0.01$), ***: significantly better than "DAE_X_1_xf" or "DAE_X_xf" ($p < 0.01$), ****: significantly better than "HCDAE_X_x+xf" ($p < 0.01$), +: significantly better than "X_2+X_3+X_4+X_5" ($p < 0.01$).

discounting. We perform pairwise ranking optimization (Hopkins and May, 2011) to tune feature weights. The translation quality is evaluated by case-insensitive IBM BLEU-4 metric.

The baseline translation models are generated by Moses with default parameter settings. In the contrast experiments, our DAE and HCDAE features are appended as extra features to the phrase table. The details of the used network structure in our experiments are shown in Table 1.

5.2 Results

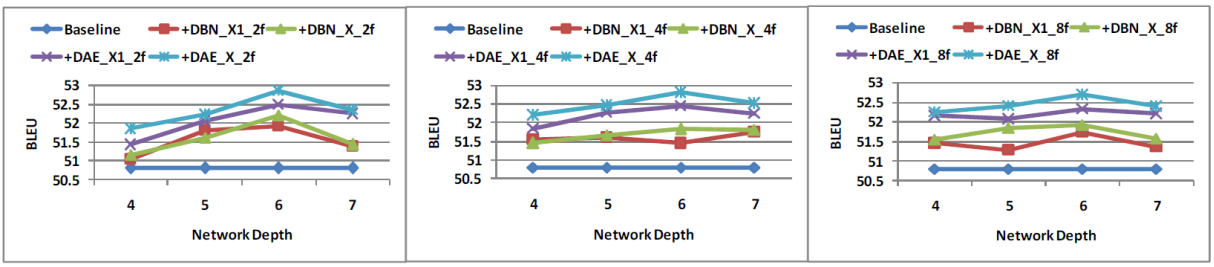
Table 2 presents the main translation results. We use DBN, DAE and HCDAE (with 6 layers' network depth), input features X_1 and X, to learn 2-, 4- and 8-dimensional features, respectively. From the results, we can get some clear trends:

1. Adding new DNN features as extra features significantly improves translation accuracy (row 2-17 vs. 1), with the highest increase of 2.45 (IWSLT) and 1.52 (NIST) (row 14 vs. 1) BLEU points over the baseline features.

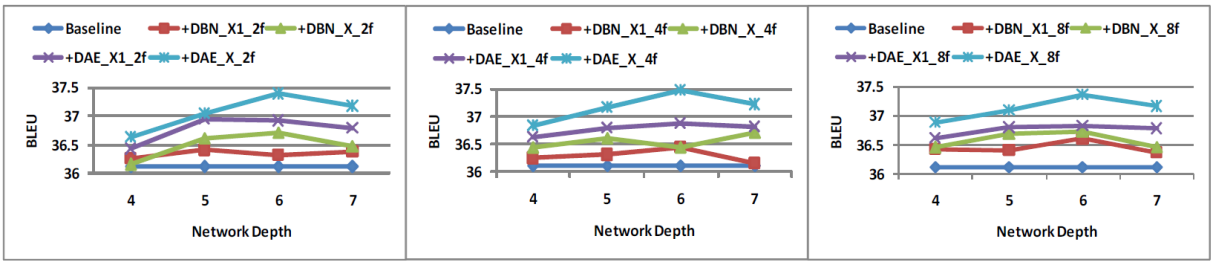
2. Compared with the unsupervised DBN features, our semi-supervised DAE features are more effective for translation decoder (row 3 vs. 2; row 5 vs. 4; row 8 vs. 7; row 11 vs. 10; row 13 vs. 12; row 16 vs. 15). Specially, Table 3 shows the variance distributions of the learned each dimensional DBN and DAE feature, our DAE features have bigger variance distributions which means

Features	IWSLT				NIST			
	σ_1	σ_2	σ_3	σ_4	σ_1	σ_2	σ_3	σ_4
DBN_X_1_4f	0.1678	0.2873	0.2037	0.1622	0.0691	0.1813	0.0828	0.1637
DBN_X_4f	0.2010	0.1590	0.2793	0.1692	0.1267	0.1146	0.2147	0.1051
DAE_X_1_4f	0.5072	0.4486	0.1309	0.6012	0.2136	0.2168	0.2047	0.2526
DAE_X_4f	0.5215	0.4594	0.2371	0.6903	0.2421	0.2694	0.3034	0.2642

Table 3: The variance distributions of each dimensional learned DBN feature and DAE feature on the two tasks.



(a) IWSLT



(b) NIST

Figure 4: The compared results of feature learning with different network structures on two development sets.

Features	IWSLT		NIST	
	Dev	Test	Dev	Test
+DAE_X_1_4f	52.45	43.06	36.88	33.47
+DAE_X_1+X_2_4f	52.76	43.38*	37.28	33.80*
+DAE_X_1+X_3_4f	52.61	43.27*	37.13	33.66*
+DAE_X_1+X_4_4f	52.52	43.24*	36.96	33.58*
+DAE_X_1+X_5_4f	52.49	43.13*	36.96	33.56*
+DAE_X_4f	52.81	43.47*	37.48	33.92*

Table 4: The effectiveness of our introduced input features. "DAE_X_1+X_i_4f" represents that we use DAE, input features $X_1 + X_i$, to learn 4-dimensional features. *: significantly better than "DAE_X_1_4f" ($p < 0.05$).

our DAE features have more discriminative power, and also their variance distributions are more stable.

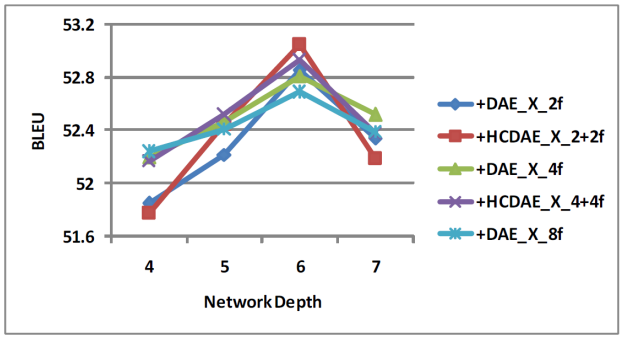
3. HCDAE outperforms single DAE for high dimensional feature learning (row 6 vs. 5; row 9 vs. 8; row 14 vs. 13; row 17 vs. 16), and further improve the performance of DAE feature learning, which can also somewhat address the bring shortcoming of the limited input features.

4. Except for the phrase feature X_1 (Maskey and Zhou, 2012), our introduced input features X significantly improve the DAE feature learning (row 11 vs. 3; row 13 vs. 5; row 16 vs. 8). Specially, Table 4 shows the detailed effectiveness of our introduced input features for DAE feature learning, and the results show that each type of features are very effective for DAE feature learning.

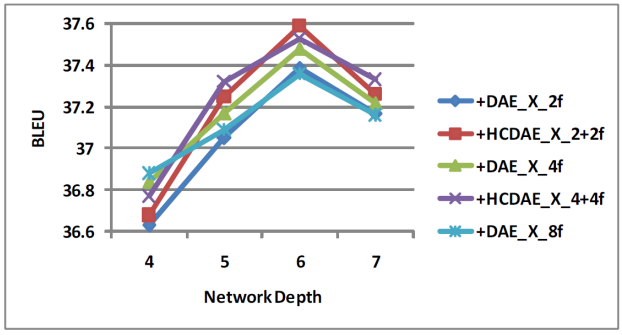
5. Adding the original features (X_2, X_3, X_4 and X_5) and DAE/HCDAE features together can further improve translation performance (row 19-23 vs. 18), with the highest increase of 3.16 (IWSLT) and 2.06 (NIST) (row 21 vs. 1) BLEU points over the baseline features. DAE and HCDAE features are learned from the non-linear combination of the original features, they strong capture high-order correlations between the activities of the original features, as to be further interpreted to reach their potential for SMT. We suspect these learned fea-

tures are complementary to the original features.

5.3 Analysis



(a) IWSLT



(b) NIST

Figure 5: The compared results of using single DAE and the HCDAE for feature learning on two development sets.

Figure 4 shows our DAE features are not only more effective but also more stable than DBN features with various network structures. Also, adding more input features (X vs. X_1) not only significantly improves the performance of DAE feature learning, but also slightly improves the performance of DBN feature learning.

Figure 5 shows there is little change in the performance of using single DAE to learn different dimensional DAE features, but the 4-dimensional features work more better and more stable. HCDAE outperforms the single DAE and learns high-dimensional representation more effectively, especially for the peak point in each condition.

Figures 5 also shows the best network depth for DAE feature learning is 6 layers. When the network depth of DBN is 7 layers, the network depth of corresponding DAE during the fine-tuning is 13 layers. Although we have pre-trained the corresponding DBN, this DAE network is so deep, the fine-tuning does not work very well and typically finds poor local minima. We suspect this leads to the decreased performance.

6 Conclusions

In this paper, instead of designing new features based on intuition, linguistic knowledge and domain, we have learned new features using the DAE for the phrase-based translation model. Using the unsupervised pre-trained DBN to initialize DAE's parameters and using the input original phrase features as the "teacher" for semi-supervised back-propagation, our semi-supervised DAE features are more effective and stable than the unsupervised DBN features (Maskey and Zhou, 2012). Moreover, to further improve the performance, we introduce some simple but effective features as the input features for feature learning. Lastly, to learn high dimensional feature representation, we introduce a natural horizontal composition of two DAEs for large hidden layers feature learning.

On two Chinese-English translation tasks, the results demonstrate that our solutions solve the two aforementioned shortcomings successfully. Firstly, our DAE features obtain statistically significant improvements of 1.34/2.45 (IWSLT) and 0.82/1.52 (NIST) BLEU points over the DBN features and the baseline features, respectively. Secondly, compared with the baseline phrase features X_1, our introduced input original phrase features X significantly improve the performance of not only our DAE features but also the DBN features.

The results also demonstrate that DNN (DAE and HCDAE) features are complementary to the original features for SMT, and adding them together obtain statistically significant improvements of 3.16 (IWSLT) and 2.06 (NIST) BLEU points over the baseline features. Compared with the original features, DNN (DAE and HCDAE) features are learned from the non-linear combination of the original features, they strong capture high-order correlations between the activities of the original features, and we believe this deep learning paradigm induces the original features to further reach their potential for SMT.

Acknowledgments

This work was supported by 863 program in China (No. 2011AA01A207). We would like to thank Xingyuan Peng, Lichun Fan and Hongyan Li for their helpful discussions. We also thank the anonymous reviewers for their insightful comments.

References

Michael Auli, Michel Galley, Chris Quirk and Geoffrey Zweig. 2013. Joint language and translation modeling with recurrent neural networks. In *Proceedings of EMNLP*, pages 1044-1054.

Pierre Baldi. 2012. Autoencoders, unsupervised learning, and deep architectures. *JMLR: workshop on unsupervised and transfer learning*, 27:37-50.

Miguel A. Carreira-Perpinan and Geoffrey E. Hinton. 2005. On contrastive divergence learning. In *Proceedings of AI and Statistics*.

George Dahl, Dong Yu, Li Deng, and Alex Acero. 2012. Context-dependent pre-trained deep neural networks for large vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):30-42.

Li Deng, Mike Seltzer, Dong Yu, Alex Acero, Abdelrahman Mohamed, and Geoffrey E. Hinton. 2010. Binary coding of speech spectrograms using a deep auto-encoder. In *Proceedings of INTERSPEECH*, pages 1692-1695.

George Foster, Cyril Goutte, and Roland Kuhn. 2010. Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of EMNLP*, pages 451-459.

Geoffrey E. Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771-1800.

Geoffrey E. Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury. 2012. Deep neural networks for acoustic modeling in speech tecognition. *IEEE Signal Processing Magazine*, 29(6):82-97.

Geoffrey E. Hinton, Alex Krizhevsky, and Sida D. Wang. 2001. Transforming auto-encoders. In *Proceedings of ANN*.

Geoffrey E. Hinton and Ruslan R. Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science*, 313:504-507.

Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527-1544.

Mark Hopkins and Jonathan May 2011. Tuning as ranking. In *Proceedings of EMNLP*, pages 1352-1362.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of EMNLP*, pages 1700-1709.

Philipp Koehn. 2004. Statistical significance tests from achine translation evaluation. In *Proceedings of ACL*, pages 388-395.

Philipp Koehn. 2010. Statistical machine translation. *Cambridge University Press*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL, Demonstration Session*, pages 177-180.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of NAACL*, pages 48-54.

Hai-Son Le, Alexandre Allauzen, and François Yvon. 2012. Continuous space translation models with neural networks. In *Proceedings of NAACL*, pages 39-48.

Peng Li, Yang Liu, Maosong Sun. 2013. Recursive autoencoders for ITG-based translation. In *Proceedings of EMNLP*, pages 567-577.

Lemao Liu, Taro Watanabe, Eiichiro Sumita, and Tiejun Zhao. 2013. Additive neural networks for statistical machine translation. In *Proceedings of ACL*, pages 791-801.

Shixiang Lu, Wei Wei, Xiaoyin Fu and Bo Xu. 2014. Recursive neural network based word topology model for hierarchical phrase-based speech translation. In *Proceedings of ICASSP*.

Yuval Marton and Philip Resnik. 2008. Soft syntactic constraints for hierarchical phrase-based translation. In *Proceedings of ACL*, pages 1003-1011.

Sameer Maskey and Bowen Zhou. 2012. Unsupervised deep belief features for speech translation. In *Proceedings of INTERSPEECH*.

Piotr Mirowski, MarcAurelio Ranzato, and Yann LeCun. 2010. Dynamic auto-encoders for semantic indexing. In *Proceedings of NIPS-2010 Workshop on Deep Learning*.

Patrick Nguyen, Milind Mahajan, and Xiaodong He. 2007. Training non-parametric features for statistical machine translation. In *Proceedings of WMT*, pages 72-79.

Franz J. Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of ACL*, pages 440-447.

Franz J. Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of ACL*, pages 295-302.

Franz J. Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417-449.

David Rumelhart, Geoffrey E. Hinton, and Ronale Williams. 1986. Learning internal representations by back-propagation errors. *Parallel Distributed Processing, Vol 1: Foundations, MIT Press*.

Ruslan R. Salakhutdinov and Geoffrey E. Hinton. 2009. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969-978.

Richard Socher, Cliff C. Lin, Andrew Y. Ng, and Christopher D. Manning. 2011. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of ICML*.

Deyi Xiong, Min Zhang, and Haizhou Li. 2011. Enhancing language models in statistical machine translation with backward n-grams and mutual information triggers. In *Proceedings of ACL*, pages 1288-1297.

Bing Zhao, Stephan Vogel, and Alex Waibel. 2004. Phrase pair rescoring with term weightings for statistical machine translation. In *Proceedings of EMNLP*.