Word Embedding and WordNet Based Metaphor Identification and Interpretation

Rui Mao, Chenghua Lin and **Frank Guerin**
Department of Computing Science
University of Aberdeen
Aberdeen, United Kingdom
{r03rm16, chenghua.lin, f.guerin}@abdn.ac.uk

Abstract

Metaphoric expressions are widespread in natural language, posing a significant challenge for various natural language processing tasks such as Machine Translation. Current word embedding based metaphor identification models cannot identify the exact metaphorical words within a sentence. In this paper, we propose an unsupervised learning method that identifies and interprets metaphors at word-level without any preprocessing, outperforming strong baselines in the metaphor identification task. Our model extends to interpret the identified metaphors, paraphrasing them into their literal counterparts, so that they can be better translated by machines. We evaluated this with two popular translation systems for English to Chinese, showing that our model improved the systems significantly.

1 Introduction

Metaphor enriches language, playing a significant role in communication, cognition, and decision making. Relevant statistics illustrate that about one third of sentences in typical corpora contain metaphor expressions (Cameron, 2003; Martin, 2006; Steen et al., 2010; Shutova, 2016). Linguistically, metaphor is defined as a language expression that uses one or several words to represent another concept, rather than taking their literal meanings of the given words in the context (Lagerwerf and Meijers, 2008). Computational metaphor processing refers to modelling non-literal expressions (e.g., metaphor, metonymy, and personification) and is useful for improving many NLP tasks such as Machine Translation (MT) and Sentiment Analysis (Rentoumi et al., 2012). For instance, Google Translate failed in translating *devour* within a sentence, *"She devoured his novels."* (Mohammad et al., 2016), into Chinese. The term was translated into 吞噬, which takes the literal sense of *swallow* and is not understandable in Chinese. Interpreting metaphors allows us to paraphrase them into literal expressions which maintain the intended meaning and are easier to translate.

Metaphor identification approaches based on word embeddings have become popular (Tsvetkov et al., 2014; Shutova et al., 2016; Rei et al., 2017) as they do not rely on hand-crafted knowledge for training. These models follow a similar paradigm in which input sentences are first parsed into phrases and then the metaphoricity of the phrases is identified; they do not tackle word-level metaphor. E.g., given the former sentence *"She devoured his novels."*, the aforementioned methods will first parse the sentence into a verb-direct object phrase *devour novel*, and then detect the clash between *devour* and *novel*, flagging this phrase as a likely metaphor. However, which component word is metaphorical cannot be identified, as important contextual words in the sentence were excluded while processing these phrases. Discarding contextual information also leads to a failure to identify a metaphor when both words in the phrase are metaphorical, but taken out of context they appear literal. E.g., *"This young man knows how to climb the social ladder."* (Mohammad et al., 2016) is a metaphorical expression. However, when the sentence is parsed into a verb-direct object phrase, *climb ladder*, it appears literal.

In this paper, we propose an unsupervised metaphor processing model which can identify and interpret linguistic metaphors at the word-level. Specifically, our model is built upon word embedding methods (Mikolov et al., 2013) and uses WordNet (Fellbaum, 1998) for lexical re-

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, pages 1–10
Melbourne, Australia, July 15 - 20, 2018. ©2018 Association for Computational Linguistics

lation acquisition. Our model is distinguished from existing methods in two aspects. First, our model is generic which does not constrain the source domain of metaphor. Second, the developed model does not rely on any labelled data for model training, but rather captures metaphor in an unsupervised, data-driven manner. Linguistic metaphors are identified by modelling the distance (in vector space) between the target word's literal and metaphorical senses. The metaphorical sense within a sentence is identified by its surrounding context within the sentence, using word embedding representations and WordNet. This novel approach allows our model to operate at the sentence level without any preprocessing, e.g., dependency parsing. Taking contexts into account also addresses the issue that a two-word phrase appears literal, but it is metaphoric within a sentence (e.g., the *climb ladder* example).

We evaluate our model against three strong baselines (Melamud et al., 2016; Shutova et al., 2016; Rei et al., 2017) on the task of metaphor identification. Extensive experimentation conducted on a publicly available dataset (Mohammad et al., 2016) shows that our model significantly outperforms the unsupervised learning baselines (Melamud et al., 2016; Shutova et al., 2016) on both phrase and sentence evaluation, and achieves equivalent performance to the state-of-the-art deep learning baseline (Rei et al., 2017) on phrase-level evaluation. In addition, while most of the existing works on metaphor processing solely evaluate the model performance in terms of metaphor classification accuracy, we further conducted another set of experiments to evaluate how metaphor processing can be used for supporting the task of MT. Human evaluation shows that our model improves the metaphoric translation significantly, by testing on two prominent translation systems, namely, Google Translate[1] and Bing Translator[2]. To our best knowledge, this is the first metaphor processing model that is evaluated on MT.

To summarise, the contributions of this paper are two-fold: (1) we proposed a novel framework for metaphor identification which does not require any preprocessing or annotated corpora for training; (2) we conducted, to our knowledge, the first metaphor interpretation study of applying metaphor processing for supporting MT. We describe related work in §2, followed by our labelling method in §4, experimental design in §5, results in §6 and conclusions in §7.

[1] https://translate.google.co.uk
[2] https://www.bing.com/translator

2 Related Work

A wide range of methods have been applied for computational metaphor processing. Turney et al. (2011); Neuman et al. (2013); Assaf et al. (2013) and Tsvetkov et al. (2014) identified metaphors by modelling the abstractness and concreteness of metaphors and non-metaphors, using a machine usable dictionary called MRC Psycholinguistic Database (Coltheart, 1981). They believed that metaphorical words would be more abstract than literal ones. Some researchers used topic models to identify metaphors. For instance, Heintz et al. (2013) used Latent Dirichlet Allocation (LDA) (Blei et al., 2003) to model source and target domains, and assumed that sentences containing words from both domains are metaphorical. Strzalkowski et al. (2013) assumed that metaphorical terms occur out of the topic chain, where a topic chain is constructed by topical words that reveal the core discussion of the text. Shutova et al. (2017) performed metaphorical concept mappings between the source and target domains in multi-languages using both unsupervised and semi-supervised learning approaches. The source and target domains are represented by semantic clusters, which are derived through the distribution of the co-occurrences of words. They also assumed that when contextual vocabularies are from different domains then there is likely to be a metaphor.

There is another line of approaches based on word embeddings. Generally, these works are not limited by conceptual domains and hand-crafted knowledge. Shutova et al. (2016) proposed a model that identified metaphors by employing word and image embeddings. The model first parses sentences into phrases which contain target words. In their word embedding based approach, the metaphoricity of a phrase was identified by measuring the cosine similarity of two component words in the phrase, based on their input vectors from Skip-gram word embeddings. If the cosine similarity is higher than a threshold, the phrase is identified as literal; otherwise metaphorical. Rei et al. (2017) identified metaphors by introducing a deep learning architecture. Instead of using word input vectors directly, they filtered out noisy in-

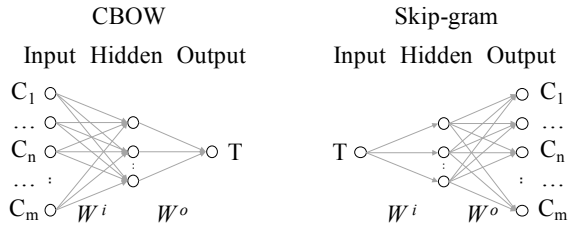

Figure 1: CBOW and Skip-gram framework.

formation in the vector of one word in a phrase, projecting the word vector into another space via a sigmoid activation function. The metaphoricity of the phrases was learnt via training a supervised deep neural network.

The above word embedding based models, while demonstrating some success in metaphor identification, only explored using input vectors, which might hinder their performance. In addition, metaphor identification is highly dependent on its context. Therefore, phrase-level models (e.g., Tsvetkov et al. (2014); Shutova et al. (2016); Rei et al. (2017)) are likely to fail in the metaphor identification task if important contexts are excluded. In contrast, our model can operate at the sentence level which takes into account rich context and hence can improve the performance of metaphor identification.

3 Preliminary: CBOW and Skip-gram

Our metaphor identification framework is built upon word embedding, which is based on Continuous Bag of Words (CBOW) and Skip-gram (Mikolov et al., 2013).

In CBOW (see Figure 1), the input and output layers are context (C) and centre word (T) one-hot encodings, respectively. The model is trained by maximizing the probability of predicting a centre word, given its context (Rong, 2014):

$$\arg\max p(t|c_1, ..., c_n, ..., c_m) \tag{1}$$

where t is a centre word, c_n is the nth context word of t within a sentence, totally m context words. CBOW's hidden layer is defined as:

$$\begin{aligned} H_{CBOW} &= \frac{1}{m} \times W^{i\top} \times \sum_{n=1}^{m} C_n \\ &= \frac{1}{m} \times \sum_{n=1}^{m} v_{c,n}^{i\top} \end{aligned} \tag{2}$$

where C_n is the one-hot encoding of the nth context word, $v_{c,n}^i$ is the nth context word row vector (input vector) in W^i which is a weight matrix between input and hidden layers. Thus, the hidden layer is the transpose of the average of input vectors of context words. The probability of predicting a centre word in its context is given by a softmax function below:

$$u_t = W_t^{o\top} \times H_{CBOW} = v_t^{o\top} \times H_{CBOW} \tag{3}$$

$$p(t|c_1, ..., c_n, ..., c_m) = \frac{\exp(u_t)}{\sum_{j=1}^{V} \exp(u_j)} \tag{4}$$

where W_t^o is equivalent to the output vector v_t^o which is essentially a column vector in a weight matrix W^o that is between hidden and output layers, aligning with the centre word t. V is the size of vocabulary in the corpus.

The output is a one-hot encoding of the centre word. W^i and W^o are updated via back propagation of errors. Therefore, only the value of the position that represents the centre word's probability, i.e., $p(t|c_1, ..., c_n, ..., c_m)$, will get close to the value of 1. In contrast, the probability of the rest of the words in the vocabulary will be close to 0 in every centre word training. W^i embeds context words. Vectors within W^i can be viewed as context word embeddings. W^o embeds centre words, vectors in W^o can be viewed as centre word embeddings.

Skip-gram is the reverse of CBOW (see Figure 1). The input and output layers are centre word and context word one-hot encodings, respectively. The target is to maximize the probability of predicting each context word, given a centre word:

$$\arg\max p(c_1, ..., c_n, ..., c_m|t) \tag{5}$$

Skip-gram's hidden layer is defined as:

$$H_{SG} = W^{i\top} \times T = v_t^{i\top} \tag{6}$$

where T is the one-hot encoding of the centre word t. Skip-gram's hidden layer is equal to the transpose of a centre word's input vector v_t, as only the tth row are kept by the operation. The probability of a context word is:

$$u_{c,n} = W_{c,n}^{o\top} \times H_{SG} = v_{c,n}^{o\top} \times H_{SG} \tag{7}$$

$$p(c_n|t) = \frac{\exp(u_{c,n})}{\sum_{j=1}^{V} \exp(u_j)} \tag{8}$$

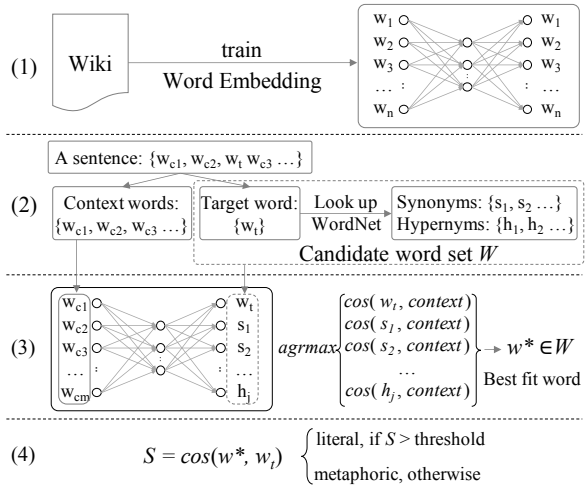

Figure 2: Metaphor identification framework. NB: $w^* =$ best fit word, $w_t =$ target word.

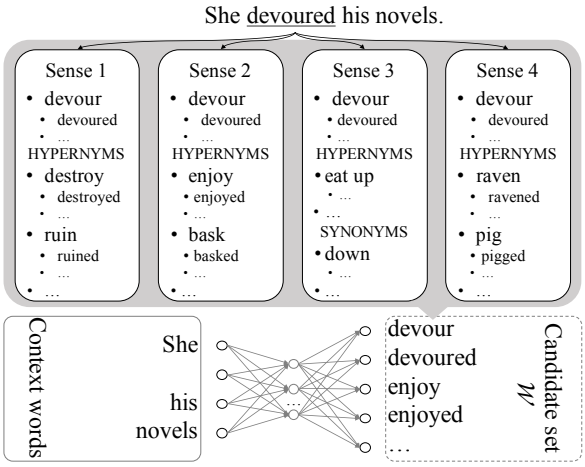

Figure 3: Given CBOW trained input and output vectors, a target word of *devoured*, and a context of *She* [ ] *his novels*, $cos(v^o_{devoured}, v^i_{context}) = -0.01$, $cos(v^o_{enjoyed}, v^i_{context}) = 0.02$.

where c, n is the nth context word, given a centre word. In Skip-gram, W^i aligns to centre words, while W^o aligns to context words. Because the names of centre word and context word embeddings are reversed in CBOW and Skip-gram, we will uniformly call vectors in W^i input vectors v^i, and vectors in W^o output vectors v^o in the remaining sections. Word embeddings represent both input and output vectors.

4 Methodology

In this section, we present the technical details of our metaphor processing framework, built upon two hypotheses. Our first hypothesis (**H1**) is that a metaphorical word can be identified, if the sense the word takes within its context and its literal sense come from different domains. Such a hypothesis is based on the theory of Selectional Preference Violation (Wilks, 1975, 1978) that a metaphorical item can be found in a violation of selectional restrictions, where a word does not satisfy its semantic constrains within a context. Our second hypothesis (**H2**) is that the literal senses of words occur more commonly in corpora than their metaphoric senses (Cameron, 2003; Martin, 2006; Steen et al., 2010; Shutova, 2016).

Figure 2 depicts an overview of our metaphor identification framework. The workflow of our framework is as follows. Step (1) involves training word embeddings based on a Wikipedia dump[3] for obtaining input and output vectors of words.

[3] https://dumps.wikimedia.org/enwiki/20170920/

In Step (2), given an input sentence, the target word (i.e., the word in the original text whose metaphoricity is to be determined) and its context words (i.e., all other words in the sentence excluding the target word) are separated. We construct a candidate word set $\mathcal{W}$ which represents all the possible senses of the target word. This is achieved by first extracting the synonyms and direct hypernyms of the target word from WordNet, and then augmenting the set with the inflections of the extracted synonyms and hypernyms, as well as the target word and its inflections. Auxiliary verbs are excluded from this set, as these words frequently appear in most sentences with little lexical meaning. In Step (3), we identify the best fit word, which is defined as the word that represents the literal sense that the target word is most likely taking given its context. Finally, in Step (4), we compute the cosine similarity between the target word and the best fit word. If the similarity is above a threshold, the target word will be identified as literal, otherwise metaphoric (i.e., based on **H1**). We will discuss in detail Step (3) and Step (4) in §4.1.

4.1 Metaphor identification

Step (3): One of the key steps of our metaphor identification framework is to identify the best fit word for a target word given its surrounding context. The intuition is that the best fit word will represent the literal sense that the target word is most likely taking. E.g., for the sentence *"She devoured his novels."* and the corresponding target word *devoured*, the best fit word is *enjoyed*, as shown in

Figure 3. Also note that the best fit word could be the target word itself if the target word is used literally.

Given a sentence s, let w_t be the target word of the sentence, $w^* \in \mathcal{W}$ the best fit word for w_t, and $w_{context}$ the surrounding context for w_t, i.e., all the words in s excluding w_t. We compute the context embedding $v^i_{context}$ by averaging out the input vectors of each context word of $w_{context}$, based on Eq. 2. Next, we rank each candidate word $k \in \mathcal{W}$ by measuring its similarity to the context input vector $v^i_{context}$ in the vector space. The candidate word with the highest similarity to the context is then selected as the best fit word.

$$w^* = \arg\max_k \text{SIM}(v_k, v_{context}) \quad (9)$$

where v_k is the vector of a candidate word $k \in \mathcal{W}$. In contrast to existing word embedding based methods for metaphor identification which only make use of input vectors (Shutova et al., 2016; Rei et al., 2017), we explore using both input and output vectors of CBOW and Skip-gram embeddings when measuring the similarity between a candidate word and the context. We expect that using a combination of input and output vectors might work better. Specifically, we have experimented with four different model variants as shown below.

$$\text{SIM-CBOW}_I = \cos(v^i_{k,cbow}, v^i_{context,cbow}) \quad (10)$$

$$\text{SIM-CBOW}_{I+O} = \cos(v^o_{k,cbow}, v^i_{context,cbow}) \quad (11)$$

$$\text{SIM-SG}_I = \cos(v^i_{k,sg}, v^i_{context,sg}) \quad (12)$$

$$\text{SIM-SG}_{I+O} = \cos(v^o_{k,sg}, v^i_{context,sg}) \quad (13)$$

Here, $\cos(\cdot)$ is cosine similarity, $cbow$ is CBOW word embeddings, sg is Skip-gram word embeddings. We have also tried other model variants using output vectors for $v_{context}$. However, we found that the models using output vectors for $v_{context}$ (both CBOW and Skip-gram embeddings) do not improve our framework performance. Due to the page limit we omitted the results of those models in this paper.

Step (4): Given a predicted best fit word w^* identified in Step (3), we then compute the cosine similarity between the lemmatizations of w^* and the target word w_t using their input vectors.

$$\text{SIM}(w^*, w_t) = \cos(v^i_{w^*}, v^i_{w_t}) \quad (14)$$

We give a detailed discussion in §4.2 of our rationale for using input vectors for Eq. 14.

If the similarity is higher than a threshold (τ) the target word is considered as literal, otherwise, metaphorical (based on **H1**). One benefit of our approach is that it allows one to paraphrase the identified metaphorical target word into the best fit word, representing its literal sense in the context. Such a feature is useful for supporting other NLP tasks such as Machine Translation, which we will explore in §6. In terms of the value of threshold (τ), it is empirically determined based on a development set. Please refer to §5 for details.

To better explain the workflow of our framework, we now go through an example as illustrated in Figure 3. The target word of the input sentence, *"She devoured his novels."* is *devoured*, and its the lemmatised form *devour* has four verbal senses in WordNet, i.e., *destroy completely*, *enjoy avidly*, *eat up completely with great appetite*, and *eat greedily*. Each of these senses has a set of corresponding synonyms and hypernyms. E.g., Sense 3 (*eat up completely with great appetite*) has synonyms *demolish*, *down*, *consume*, and hypernyms *go through*, *eat up*, *finish*, and *polish off*. We then construct a candidate word set $\mathcal{W}$ by including the synonyms and direct hypernyms of the target word from WordNet, and then augmenting the set with the inflections of the extracted synonyms and hypernyms, as well as the target word *devour* and its inflections. We then identify the best fit word given the context *she* [ ] *his novels* based on Eq. 9. Based on **H2**, literal expressions are more common than metaphoric ones in corpora. Therefore, the best fit word is expected to frequently appear within the given context, and thus represents the most likely sense of the target word. For example, the similarity between *enjoy* (i.e., the best fit word) and the the context is higher than that of *devour* (i.e., the target word), as shown in Figure 3.

4.2 Word embedding: output vectors vs. input vectors

Typically, input vectors are used after training CBOW and Skip-gram, with output vectors being abandoned by practical models, e.g., original word2vec model (Mikolov et al., 2013) and Gensim toolkit (Řehůřek and Sojka, 2010), as these models are designed for modelling similarities in semantics. However, we found that using input vectors to measure cosine similarity between two words with different POS types in a phrase is sub-

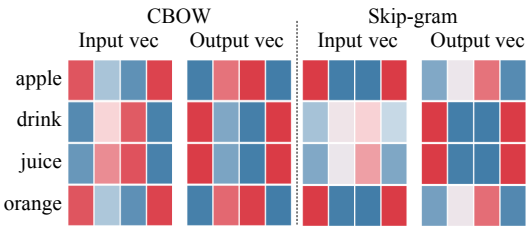

Figure 4: Input and output vector visualization. The bluer, the more negative. The redder, the more positive.

optimal, as words with different POS normally have different semantics. They tend to be distant from each other in the input vector space. Taking Skip-gram for example, empirically, input vectors of words with the same POS, occurring within the same contexts tend to be close in the vector space (Mikolov et al., 2013), as they are frequently updated by back propagating the errors from the same context words. In contrast, input vectors of words with different POS, playing different semantic and syntactic roles tend to be distant from each other, as they seldom occur within the same contexts, resulting in their input vectors rarely being updated equally. Our observation is also in line with Nalisnick et al. (2016), who examine IN-IN, OUT-OUT and IN-OUT vectors to measure similarity between two words. Nalisnick et al. discovered that two words which are similar by function or type have higher cosine similarity with IN-IN or OUT-OUT vectors, while using input and output vectors for two words (IN-OUT) that frequently co-occur in the same context (e.g., a sentence) can obtain a higher similarity score.

For illustrative purpose, we visualize the CBOW and Skip-gram updates between 4-dimensional input and output vectors by Wevi[4] (Rong, 2014), using a two-sentence corpus, *"Drink apple juice."* and *"Drink orange juice."*. We feed these two sentences to CBOW and Skip-gram with 500 iterations. As seen Figure 4, the input vectors of *apple* and *orange* are similar in both CBOW and Skip-gram, which are different from the input vectors of their context words (*drink* and *juice*). However, the output vectors of *apple* and *orange* are similar to the input vectors of *drink* and *juice*.

To summarise, using input vectors to compare similarity between the best fit word and the target word is more appropriate (cf. Eq.14), as they tend to have the same types of POS. When measuring the similarity between candidate words and the context, using output vectors for the former and input vectors for the latter seems to better predict the best fit word.

5 Experimental settings

Baselines. We compare the performance of our framework for metaphor identification against three strong baselines, namely, an unsupervised word embedding based model by Shutova et al. (2016), a supervised deep learning model by Rei et al. (2017), and the Context2Vec model[5] (Melamud et al., 2016) which achieves the best performance on Microsoft Sentence Completion Challenge (Zweig and Burges, 2011). Context2Vec was not designed for processing metaphors, in order to use it for this we plug it into a very similar framework to that described in Figure 2. We use Context2Vec to predict the best fit word from the candidate set, as it similarly uses context to predict the most likely centre word but with bidirectional LSTM based context embedding method. After locating the best fit word with Context2Vec, we identify the metaphoricity of a target word with the same method (see Step (4) in §4), so that we can also apply it for metaphor interpretation. Note that while Shutova et al. and Rei et al. detect metaphors at the phrase level by identifying metaphorical phrases, Melamud et al.'s model can perform metaphor identification and interpretation on sentences.

Dataset. Evaluation was conducted based on a dataset developed by Mohammad et al. (2016). This dataset[6], containing 1,230 literal and 409 metaphor sentences, has been widely used for metaphor identification related research (Shutova et al., 2016; Rei et al., 2017). There is a verbal target word annotated by 10 annotators in each sentence. We use two subsets of the Mohammad et al. set, one for phrase evaluation and one for sentence evaluation. The phrase evaluation dataset was kindly provided by Shutova, which consists of 316 metaphorical and 331 literal phrases (subject-verb and verb-direct object word pairs), parsed from Mohammad et al.'s dataset. Similar to Shutova et al. (2016), we use 40 metaphoric and 40 literal phrases as a development set and the rest as a test

[4] https://ronxin.github.io/wevi/

[5] http://u.cs.biu.ac.il/~nlp/resources/downloads/context2vec/

[6] http://saifmohammad.com/WebPages/metaphor.html

	Method	P	R	F1
Phrase	Shutova et al. (2016)	0.67	0.76	0.71
	Rei et al. (2017)	**0.74**	0.76	**0.74**
	SIM-CBOW$_{I+O}$	0.66	0.78	0.72
	SIM-SG$_{I+O}$	0.68	**0.82**	**0.74***
Sent.	Melamud et al. (2016)	0.60	0.80	0.69
	SIM-SG$_{I}$	0.56	**0.95**	0.70
	SIM-SG$_{I+O}$	0.62	0.89	0.73
	SIM-CBOW$_{I}$	0.59	0.91	0.72
	SIM-CBOW$_{I+O}$	**0.66**	0.88	**0.75***

Table 1: Metaphor identification results. NB: * denotes that our model outperforms the baseline significantly, based on two-tailed paired t-test with $p < 0.001$.

set.

For sentence evaluation, we select 212 metaphorical sentences whose target words are annotated with at least 70% agreement. We also add 212 literal sentences with the highest agreement. Among the 424 sentences, we form our development set with 12 randomly selected metaphoric and 12 literal instances to identify the threshold for detecting metaphors. The remaining 400 sentences are our testing set.

Word embedding training. We train CBOW and Skip-gram models on a Wikipedia dump with the same settings as Shutova et al. (2016) and Rei et al. (2017). That is, CBOW and Skip-gram models are trained iteratively 3 times on Wikipedia with a context window of 5 to learn 100-dimensional word input and output vectors. We exclude words with total frequency less than 100. 10 negative samples are randomly selected for each centre word training. The word down-sampling rate is 10^{-5}. We use Stanford CoreNLP (Manning et al., 2014) lemmatized Wikipedia to train word embeddings for phrase level evaluation, which is in line with Shutova et al. (2016). In sentence evaluation, we use the original Wikipedia for training word embeddings.

6 Experimental Results

6.1 Metaphor identification

Table 1 shows the performance of our model and the baselines on the task of metaphor identification. All the results for our models are based on a threshold of 0.6, which is empirically determined based on the developing set. For sentence level metaphor identification, it can be observed that all our models outperform the baseline (Melamud et al., 2016), with SIM-CBOW$_{I+O}$ giving the highest F1 score of 75% which is a 6% gain over the baseline. We also see that models based on both input and output vectors (i.e., SIM-CBOW$_{I+O}$ and SIM-SG$_{I+O}$) yield better performance than the models based on input vectors only (i.e., SIM-CBOW$_{I}$ and SIM-SG$_{I}$). Such an observation supports our assumption that using input and output vectors can better model similarity between words that have different types of POS, than simply using input vectors. When comparing CBOW and Skip-gram based models, we see that CBOW based models generally achieve better performance in precision whereas Skip-gram based models perform better in recall.

In terms of phrase level metaphor identification, we compare our best performing models (i.e., SIM-CBOW$_{I+O}$ and SIM-SG$_{I+O}$) against the approaches of Shutova et al. (2016) and Rei et al. (2017). In contrast to the sentence level evaluation in which SIM-CBOW$_{I+O}$ gives the best performance, SIM-SG$_{I+O}$ performs best for the phrase level evaluation. This is likely due to the fact that Skip-gram is trained by using a centre word to maximise the probability of each context word, whereas CBOW uses the average of context word input vectors to maximise the probability of the centre word. Thus, Skip-gram performs better in modelling one-word context, while CBOW has better performance in modelling multi-context words. When comparing to the baselines, our model SIM-SG$_{I+O}$ significantly outperforms the word embedding based approach by Shutova et al. (2016), and gives the same performance as the deep supervised method (Rei et al., 2017) which requires a large amount of labelled data for training and cost in training time.

SIM-CBOW$_{I+O}$ and SIM-SG$_{I+O}$ are also evaluated with different thresholds for both phrase and sentence level metaphor identification. As can be seen from Table 2, the results are fairly stable when the threshold is set between 0.5 and 0.9 in terms of F1.

6.2 Metaphor processing for MT

We believe that one of the key purposes of metaphor processing is to support other NLP tasks. Therefore, we conducted another set of experiments to evaluate how metaphor processing can be used to support English-Chinese machine translation.

The evaluation task was designed as follows. From the test set for sentence-level metaphor identification which contains 200 metaphoric and

τ	Sentence			Phrase	
	P	R	F1	$F1_{\text{SIM-CBOW}_{I+O}}$	$F1_{\text{SIM-SG}_{I+O}}$
0.3	0.75	0.60	0.67	0.56	0.46
0.4	0.69	0.75	0.72	0.65	0.63
0.5	0.67	0.82	0.74	0.71	0.72
0.6	0.66	0.88	**0.75**	**0.72**	**0.74**
0.7	0.64	0.88	0.74	0.72	0.73
0.8	0.63	0.89	0.74	0.72	0.73
0.9	0.63	0.89	0.74	0.71	0.73
1.0	0.50	1.00	0.67	0.65	0.65

Table 2: Model performance vs. different threshold (τ) settings. NB: the sentence level results are based on SIM-CBOW_{I+O}.

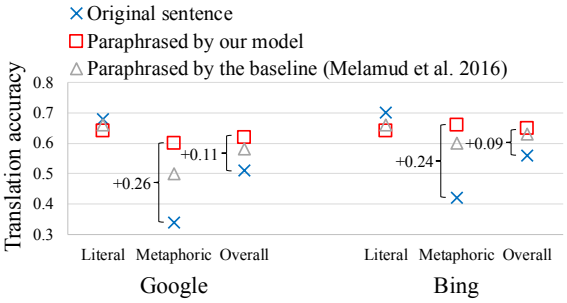

Figure 5: Accuracy of metaphor interpretation, evaluated on Google and Bing Translation.

200 literal sentences, we randomly selected 50 metaphoric and 50 literal sentences to construct a set $\mathcal{S}_{\mathcal{M}}$ for the Machine Translation (MT) evaluation task. For each sentence in $\mathcal{S}_{\mathcal{M}}$, if it is predicted as literal by our model, the sentence is kept unchanged; otherwise, the target word of the sentence is paraphrased with the best fit word (refer to §4.1 for details). The metaphor identification step resulted in 42 True Positive (TP) instances where the ground truth label is metaphoric and 19 False Positive (FP) instances where the ground truth label is literal, resulting in a total of 61 instances predicted as metaphorical by our model. We also run one of our baseline models, Context2Vec, on the 61 sentences to predict the best fit words for comparison. Our hypothesis is that by paraphrasing the metaphorically used target word with the best fit word which expresses the target word's real meaning, the performance of translation engines can be improved.

We test our hypothesis on two popular English-Chinese MT systems, i.e., the Google and Bing Translators. We recruited from a UK university 5 Computing Science postgraduate students who are Chinese native speakers to participate the English-Chinese MT evaluation task. During the evaluation, subjects were presented with a questionnaire

Sample Questionnaire

The ex-boxer's job is to **bounce** people who want to enter this private club.

bounce: eject from the premises Good / Bad

1. 前拳击手的工作是**反弹**人谁想要进入这个私人俱乐部。 ☐ ☐
2. 前拳击手的工作是让想要进入这个私人俱乐部的人**弹跳**。 ☐ ☐
3. 前拳击手的工作是**拒绝**谁想要进入这个私人俱乐部的人。 ☐ ☐
4. 前拳击手的工作是**拒绝**那些想进入这个私人俱乐部的人。 ☐ ☐
5. 前拳击手的工作是**打**人谁想要进入这个私人俱乐部。 ☐ ☐
6. 前拳击手的工作是**打击**那些想进入这个私人俱乐部的人。 ☐ ☐

Figure 6: MT-based metaphor interpretation questionnaire.

		Acc-met.	Acc-lit.	Acc-overall
Google	Orig. Sent.	0.34	**0.68**	0.51
	Context2Vec	0.50	0.66	0.58
	SIM-CBOW_{I+O}	**0.60**	0.64	**0.62**
Bing	Orig. Sent.	0.42	**0.70**	0.56
	Context2Vec	0.60	0.66	0.63
	SIM-CBOW_{I+O}	**0.66**	0.64	**0.65**

Table 3: Accuracy of metaphor interpretation, evaluated on Google and Bing Translation.

containing English-Chinese translations of each of the 100 randomly selected sentences. For each sentence predicted as literal (39 out of 100 sentences), there are two corresponding translations by Google and Bing respectively. For each sentence predicted as metaphoric (61 out of 100 sentences), there are 6 corresponding translations.

An example of the evaluation task is shown in Figure 6, in which *"The ex-boxer's job is to bounce people who want to enter this private club."* is the original sentence, followed by an WordNet explanation of the target word of the sentence (i.e., bounce: eject from the premises). There are 6 translations. No. 1-2 are the original sentence translations, translated by Google Translate (GT) and Bing Translator (BT). The target word, *bounce*, is translated, taking the sense of (1) *physically rebounding like a ball* (反弹), (2) *jumping* (弹跳). No. 3-4 are SIM-CBOW_{I+O} paraphrased sentences, translated by GT and BT, respectively, taking the sense of *refusing* (拒绝). No. 5-6 are Context2Vec paraphrased sentences, translated by GT and BT, respectively, taking the sense of *hitting* (5.打; 6.打击).

Subjects were instructed to determine if the translation of a target word can correctly represent its sense within the translated sentence, matching its context (cohesion) in Chinese. Note that we evaluate the translation of the target word, therefore, errors in context word translations are ignored by the subjects. Finally, a label is taken agreed by more than half annotators. Noticeably,

based on our observation, there is always a Chinese word corresponding to an English target word in MT, as the annotated target word normally represents important information in the sentence in the applied dataset.

We use translation accuracy as a measure to evaluate the improvement on MT systems after metaphor processing. The accuracy is calculated by dividing the number of correctly translated instances by the total number of instances. As can be seen in Figure 5 and Table 3, after paraphrasing the metaphorical sentences with the SIM-CBOW$_{I+O}$ model, the translation improvement for the metaphorical class is dramatic for both MT systems, i.e., 26% improvement for Google Translate and 24% for Bing Translate. In terms of the literal class, there is some small drop (i.e., 4-6%) in accuracy. This is due to the fact that some literals were wrongly identified as metaphors and hence error was introduced during paraphrasing. Nevertheless, with our model, the overall translation performance of both Google and Bing Translate are significantly improved by 11% and 9%, respectively. Our baseline model Context2Vec also improves the translation accuracy, but is 2-4 % lower than our model in terms of overall accuracy. In summary, the experimental results show the effectiveness of applying metaphor processing for supporting Machine Translation.

7 Conclusion

We proposed a framework that identifies and interprets metaphors at word-level with an unsupervised learning approach. Our model outperforms the unsupervised baselines in both sentence and phrase evaluations. The interpretation of the identified metaphorical words given by our model also contributes to Google and Bing translation systems with 11% and 9% accuracy improvements.

The experiments show that using words' hypernyms and synonyms in WordNet can paraphrase metaphors into their literal counterparts, so that the metaphors can be correctly identified and translated. To our knowledge, this is the first study that evaluates a metaphor processing method on Machine Translation. We believe that compared with simply identifying metaphors, metaphor processing applied in practical tasks, can be more valuable in the real world. Additionally, our experiments demonstrate that using a candidate word output vector instead of its input vector to model the similarity between the candidate word and its context yields better results in the best fit word (the literal counterpart of the metaphor) identification.

CBOW and Skip-gram do not consider the distance between a context word and a centre word in a sentence, i.e., context word contributes to predict the centre word equally. Future work will introduce weighted CBOW and Skip-gram to learn positional information within sentences.

Acknowledgments

This work is supported by the award made by the UK Engineering and Physical Sciences Research Council (Grant number: EP/P005810/1).

References

Dan Assaf, Yair Neuman, Yohai Cohen, Shlomo Argamon, Newton Howard, Mark Last, Ophir Frieder, and Moshe Koppel. 2013. Why "dark thoughts" aren't really dark: A novel algorithm for metaphor identification. In *Computational Intelligence, Cognitive Algorithms, Mind, and Brain (CCMB), 2013 IEEE Symposium on*. IEEE, pages 60–65.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent Dirichlet allocation. *Journal of machine Learning research* 3(Jan):993–1022.

Lynne Cameron. 2003. *Metaphor in educational discourse*. A&C Black.

Max Coltheart. 1981. The MRC psycholinguistic database. *The Quarterly Journal of Experimental Psychology* 33(4):497–505.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Ilana Heintz, Ryan Gabbard, Mahesh Srinivasan, David Barner, Donald S Black, Marjorie Freedman, and Ralph Weischedel. 2013. Automatic extraction of linguistic metaphor with LDA topic modeling. In *Proceedings of the First Workshop on Metaphor in NLP (ACL 2013)*. pages 58–66.

Luuk Lagerwerf and Anoe Meijers. 2008. Openness in metaphorical and straightforward advertisements: Appreciation effects. *Journal of Advertising* 37(2):19–30.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. pages 55–60.

James H Martin. 2006. A corpus-based analysis of context effects on metaphor comprehension. Technical Report CU-CS-738-94, Boulder: University of Colorado: Computer Science Department.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional LSTM. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL 2016)*. pages 51–61.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Proceedings of International Conference on Learning Representations (ICLR 2013)* .

Saif M Mohammad, Ekaterina Shutova, and Peter D Turney. 2016. Metaphor as a medium for emotion: An empirical study. *Proceedings of the Joint Conference on Lexical and Computational Semantics (*SEM 2016)* page 23.

Eric Nalisnick, Bhaskar Mitra, Nick Craswell, and Rich Caruana. 2016. Improving document ranking with dual word embeddings. In *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, pages 83–84.

Yair Neuman, Dan Assaf, Yohai Cohen, Mark Last, Shlomo Argamon, Newton Howard, and Ophir Frieder. 2013. Metaphor identification in large texts corpora. *PloS one* 8(4):e62343.

Radim Řehůřek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, pages 45–50. http://is.muni.cz/publication/884893/en.

Marek Rei, Luana Bulat, Douwe Kiela, and Ekaterina Shutova. 2017. Grasping the finer point: A supervised similarity network for metaphor detection. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)* pages 1537–1546.

Vassiliki Rentoumi, George A Vouros, Vangelis Karkaletsis, and Amalia Moser. 2012. Investigating metaphorical language in sentiment analysis: A sense-to-sentiment perspective. *ACM Transactions on Speech and Language Processing (TSLP)* 9(3):6.

Xin Rong. 2014. word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738* .

Ekaterina Shutova. 2016. Design and evaluation of metaphor processing systems. *Computational Linguistics* .

Ekaterina Shutova, Douwe Kiela, and Jean Maillard. 2016. Black holes and white rabbits: Metaphor identification with visual features. *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2016)* pages 160–170.

Ekaterina Shutova, Lin Sun, Elkin Darío Gutiérrez, Patricia Lichtenstein, and Srini Narayanan. 2017. Multilingual metaphor processing: Experiments with semi-supervised and unsupervised learning. *Computational Linguistics* 43(1):71–123.

Gerard J Steen, Aletta G Dorst, J Berenike Herrmann, Anna Kaal, Tina Krennmayr, and Trijntje Pasma. 2010. *A method for linguistic metaphor identification: From MIP to MIPVU*, volume 14. John Benjamins Publishing.

Tomek Strzalkowski, George Aaron Broadwell, Sarah Taylor, Laurie Feldman, Samira Shaikh, Ting Liu, Boris Yamrom, Kit Cho, Umit Boz, Ignacio Cases, et al. 2013. Robust extraction of metaphor from novel data. In *Proceedings of the First Workshop on Metaphor in NLP (ACL 2013)*. pages 67–76.

Yulia Tsvetkov, Leonid Boytsov, Anatole Gershman, Eric Nyberg, and Chris Dyer. 2014. Metaphor detection with cross-lingual model transfer. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)* pages 248–258.

Peter D Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. 2011. Literal and metaphorical sense identification through concrete and abstract context. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)* pages 680–690.

Yorick Wilks. 1975. A preferential, pattern-seeking, semantics for natural language inference. *Artificial Intelligence* 6(1):53–74.

Yorick Wilks. 1978. Making preferences more active. *Artificial Intelligence* 11(3):197–223.

Geoffrey Zweig and Christopher JC Burges. 2011. The Microsoft research sentence completion challenge. Technical report, Technical Report MSR-TR-2011-129, Microsoft.