How to Avoid Sentences *Spelling* Boring? Towards a Neural Approach to Unsupervised Metaphor Generation

Zhiwei Yu and **Xiaojun Wan**
Center for Data Science, Peking University
Institute of Computer Science and Technology, Peking University
The MOE Key Laboratory of Computational Linguistics, Peking University
{yuzw,wanxiaojun}@pku.edu.cn

Abstract

Metaphor generation attempts to replicate human creativity with language, which is an attractive but challengeable text generation task. Previous efforts mainly focus on template-based or rule-based methods and result in a lack of linguistic subtlety. In order to create novel metaphors, we propose a neural approach to metaphor generation and explore the shared inferential structure of a metaphorical usage and a literal usage of a verb. Our approach does not require any manually annotated metaphors for training. We extract the metaphorically used verbs with their metaphorical senses in an unsupervised way and train a neural language model from wiki corpus. Then we generate metaphors conveying the assigned metaphorical senses with an improved decoding algorithm. Automatic metrics and human evaluations demonstrate that our approach can generate metaphors with good readability and creativity.

1 Introduction

Metaphor is a kind of language filled with vitality and elasticity. It employs words in a way that deviates from their normal meaning to represent another concept (Li and Sporleder, 2010). Using metaphor allows us to express not just information but also real feelings and complex attitudes. There is a clear need in computational metaphor generation whose insightful results can be used in many applications from entertainment to education (Veale, 2016). Besides, a unified metaphor annotation procedure and creation of a large publicly available metaphor corpus are in great demands. Such resources make it possible to interpretate the identified metaphorical expressions and enhance the performance on other Natural Language Processing (NLP) applications (Shutova, 2010).

Although metaphor has a long history of academic studies in both philosophy and linguistics (Genesereth, 1980; G. and M., 1985), it still remains a tough problem for the NLP community. As the metaphor is hardly to be well-defined and modeled, little work focuses on the metaphor generation. Most of the previous efforts rely on hand-coded knowledge (Martin, 1990; Feldman and Narayanan, 2004; Agerri et al., 2007) , which heavily constrains the diversity of generated metaphors.

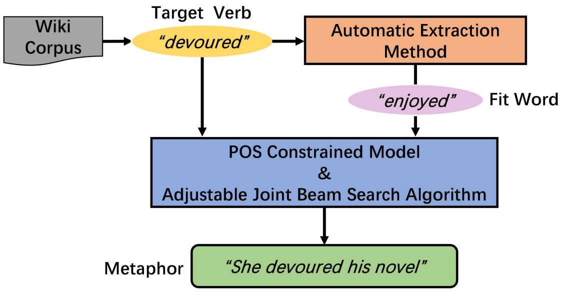
Wiki Corpus
Target Verb
"devoured"
Automatic Extraction Method
"enjoyed"
Fit Word
POS Constrained Model & Adjustable Joint Beam Search Algorithm
Metaphor
"She devoured his novel"

Figure 1: The Metaphor Generation Process

The end-to-end approach presented to sequence learning has been proved effective on the generation tasks like machine translation (Sutskever et al., 2014), abstractive summarization (Tan et al., 2017), product review generation (Zang and Wan, 2017) and multi-label classification (Yang et al., 2018). The approach is able to train a language model which can generate fluent and creative sentences with sufficient corpus. Unfortunately, in spite of the industrious exploration of the metaphor corpus, the annotated data available now is still far from training a good language model. To the best of our knowledge, there has been no work combining metaphor generation with the end-to-end approach.

In this paper, we propose a neural approach for metaphor generation trained with Wiki corpus rather than the limited annotated metaphor corpus, which assures the quality of the language model. The approach is shown in Figure 1. Relevant sta-

Proceedings of NAACL-HLT 2019, pages 861–871
Minneapolis, Minnesota, June 2 - June 7, 2019. ©2019 Association for Computational Linguistics

tistics demonstrate that the most frequent type of metaphor is expressed by a verb (Martin, 2006; Steen, 2010). In this paper, we focus on generating verb-oriented metaphors. We use an unsupervised method to extract the metaphorically used verbs in Wiki corpus. A metaphorical pair consists of a target verb (e.g. "devoured") and a fit word (e.g. "enjoyed"). And it is used to model the metaphorical usage of the target verb. According to Narayanan (1997), a metaphorical usage and a literal usage share inferential structure. We follow this intuition to find an intersection between the metaphorical usage and the literal usage of a word. For example, in "she devoured (enjoyed) his novels", the literal sense of "enjoyed" represents the contextual sense of "devoured" in such contexts. But the similarity between "enjoyed" and "devoured" is low, which means the target verb "devoured" is merely used in such sense and can be seen as a metaphorically used verb.

For metaphor generation, we first propose a POS constrained language model to generate a sentence containing a given verb and use an elaborately designed algorithm to consider its fit word simultaneously while decoding. For a profound exploration, we introduce automatic metrics as well as manual ways to evaluate the generation results. Experimental results demonstrate that our approach is capable of generating fluent and seemly metaphors. All the generated metaphors are novel and do not exist in the corpus.

To summarise, the contributions of our work are as follows[1]:

- As far as we know, our work is the first endeavor to adopt the end-to-end framework on metaphor generation. Besides, the proposed method does not require any manually labeled metaphor corpus.

- We automatically extract the verbs and their fit words in the corpus in an unsupervised way and use them (e.g. "devoured", "enjoyed") to model the metaphorical senses of the verbs for the generation process. We further explore the semantic shift of a verb by changing the adjustable factors.

- Both automatic metrics and human evaluation results demonstrate the efficacy of our model. Our model outperforms the baseline models on 3 aspects significantly[2].

2 Related Work

Metaphor is highly frequent in language and its computational processing is indispensable for real-world NLP applications addressing semantic tasks. Automatic processing of metaphor can be clearly divided into two subtasks: metaphor identification and metaphor interpretation (Shutova, 2010), little research has been devoted to the metaphor generation. In this subsection, we briefly review some prior work on metaphor generation.

Jones (1992) aims to generate a specific class of metaphors: Transparently-Motivated Metaphor, which is based on universal groundings that are often linked to bodily experience. Abe et al. (2006), Terai and Nakagawa (2010) generate metaphors in the form of "A (target) like B (vehicle)". They firstly compute the probabilistic relationship between concepts and words with a statistical analysis of language data and then select candidates to fill in the template. From a given mapping between the concepts of two domains, Hervás et al. (2007) present an approach to the application of metaphors for referring to concepts in an automatically generated text. Mason (2004) obtains domain-specific selectional preferences of verbs from large corpora and maps their common nominal arguments in other domains. The corresponding metaphorical mappings are achieved by such systematic variations and can generate simple conceptual metaphors in the form of: "A is (a) B". Ovchinnikova et al. (2014) also rely on characteristic predicate but use general propositions instead of the verb and adjective phrases. As some metaphors' target domain is lexically divorced from the source, Lederer (2016) identifies constellations of source-domain triggers in limited source domains. To make the metaphor generators more comprehensible and forceful, Veale (2016) presents a knowledge-base to generate XYZ metaphors such as "Bruce Wayne is the Donald Trump of Gotham City".

Previous methods make groundbreaking explorations on metaphor generation. However, these approaches mainly focus on modeling the phrase-level metaphor expressions and the generation process depends on the templates, which causes the lack of linguistic subtlety to some extent and

[1] https://github.com/ArleneYuZhiwei/Metaphor-Generation

[2] Based on two-tailed paired t-test with $p<0.05$.

they are not able to build a large publicly available metaphor corpus for further study. Our approach focuses on generating sentence-level metaphor in a template-free way.

3 Our Approach

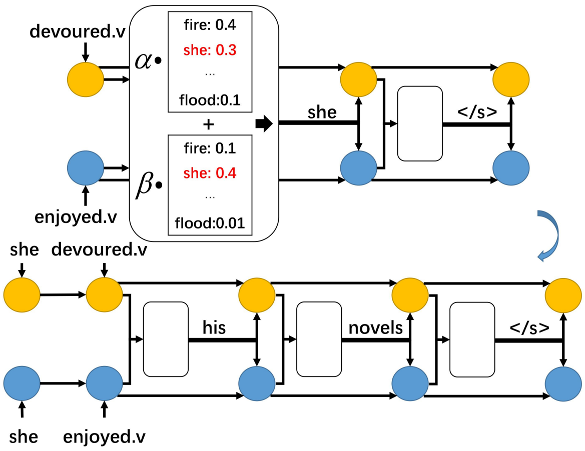

Figure 2: Adjustable Joint Model. Top: A metaphorical pair (e.g. "*devoured*" and "*enjoyed*") are given to the backward model, to generate the backward sequence. Bottom: Input the reversed backward sequence to the forward model, to generate the forward sequence. The inputs and outputs of the forward model are concatenated to form a metaphor.

Metaphors are ubiquitous in the normal corpus but lack of annotation. First, we extract the metaphorical pair which consists of a target word (e.g. "devoured") and its fit word (e.g. "enjoyed") automatically. Then we adopt an end-to-end neural framework to train a POS constrained language model which can generate a sentence containing the assigned verb. For metaphor inference, we apply an adjustable joint beam search algorithm to the decoding phase. In this way, the target verb is metaphorically used in the generated sentence. The proposed model is named Adjustable Joint Model and is shown in Figure 2.

3.1 Automatic Extraction of Metaphor Pairs

Our automatic extraction method is based on the hypotheses as follows:

H1. A metaphorical word is employed in the sentence to represent another concept and deviates from its normal meaning (Wilks, 1978; Li and Sporleder, 2010; Mao et al., 2018).

H2. The metaphorical senses of words occur with relatively lower frequency in the corpus than their literal senses do (Cameron, 2003; Martin, 2006; Mao et al., 2018).

Algorithm 1 Automatic Extraction

```
Require: $S$: the corpus, a set of sentences containing the target words
Require: $T$: the set of target words in $S$
Require: $M$: a trained CBOW model. $v_a^i$denotes the input vector of word $a$. $v_a^o$denotes the output vector of word $a$. $v_{context}^i$denotes the average input vectors of the words in the $context$.
Require: $\varepsilon$: the threshold that determines the metaphoricity.
$Inflect(C)$ gets the inflections of each word in the set $C$. $Sim(x, y)$ calculates the cosine similarity between two vectors $x, y$.
$P = \emptyset$
For any sentence $s \in S$ and the contained target word $t \in T$
$context =$ the set consists of the words in $s$ excluding $t$
$syn =$ the set consists of the synonyms of $t$
$hyp =$ the set consists of the hypernyms of $t$
$candidates = Inflect(syn \cup hyp \cup t)$
$w = \underset{k}{argmax} Sim(v_k^o, v_{context}^i), k \in candidates$
$sim = Sim(v_w^i, v_t^i)$
if $sim < \varepsilon$ then:
    $P \leftarrow P \cup (t, w)$
return $P$
```

As metaphors begin their lives with marked rhetorical effects, whose comprehension requires a special imaginative leap (Nunberg, 1987), it is intuitive to assume that a metaphorical word can be distinguished from the literal one in the corpus with the violation of semantic constraints within a context. It has been proved that the dissimilarity between neural embeddings of the two words in a phrase is indicative of identifying the metaphoricity of the phrase (Shutova et al., 2016; Rei et al., 2017). Thus we find the semantic violation in the corpus based on word embeddings. The word embeddings are obtained by using Continuous Bag-of-Words Model(CBOW) (Mikolov et al., 2013).

Inspired by Mao et al. (2018), we use a fit word to model the contextual sense of the target word. To find a fit word for the target word t, we construct a candidate word set $candidates$ which consists of the target word as well as its synonyms and direct hypernyms extracted from WordNet(Miller, 1998). The target word may have several senses. Each of these senses has a set of corresponding synonyms and hypernyms. We extract the synonyms and hypernyms of all the verb senses. We augment the set with the inflections of the items in it. The word in the $candidates$ which has the highest similarity with the given context represents the contextual sense of the target verb in the sentence.

For example, "*i am afraid this spells trouble*" is a sentence in the corpus and the target word t is "*spells*". The word (e.g. "*means*") in $candidates$ has the highest similarity with the given context (e.g. "*i am afraid this [ ] trouble.*"), and is the fit word of t.

We then compute the similarity between the target verb (e.g. "*spells*") and its fit word (e.g. "*means*") . If the similarity is less than or equal to a threshold $\varepsilon = 0.6$[3], we extract the metaphorical word together with the fit word to form a metaphorical pair. The extraction process is described in Algorithm 1.

Follow the previous work (Nalisnick et al., 2016; Mao et al., 2018), we use OUT-IN vectors to measure the similarity between a fit word and its given context, use IN-IN vectors to measure the similarity between two words in a metaphorical pair[4] .

3.2 POS Constrained Language Model

Our goal is to generate a sentence containing a metaphorically used verb. However, the vanilla end-to-end model cannot guarantee the target word appearing in the generated sequence all the time, letting alone the appearance of a word with a specific part-of-speech. To solve this problem, we present a neural language model which can ensure an assigned verb to appear in the generated sentence. Our design is inspired by the asynchronous forward/backward generation model proposed by (Mou et al., 2016).

The POS constrained language model is trained end-to-end. Given a target verb as input, the model first generates the backward sequence starting from the target word w_t at position t of the sentence and ending up with "</s>" at position 0 of the sentence. n is the position of the last word in the sentence. $p(w_t^1)$ denotes the probability of the backward sequence. Then we reverse the output of the backward sequence as the input to the forward model. And it generates the rest part of the sentence accordingly. $p(w_t^n)$ denotes the probability of the forward sequence. The generated sentence is a connection of the input and output of the forward model whose probability can be decomposed as:

$$p(\overleftarrow{w_t^1}||\overrightarrow{w_t^n}){=}p(w_t)\prod_{i=1}^{t-1}p^{(bw)}(\overleftarrow{w_{t-i}}|\cdot)\prod_{i=1}^{n-t}p^{(fw)}(\overrightarrow{w_{t+i}}|\cdot), \tag{1}$$

where $p(\cdot||\cdot)$ means the probability of a particular backward/forward sequence. $p^{(bw)}(\overleftarrow{w_t}|\cdot)$ or $p^{(fw)}(\overrightarrow{w_t}|\cdot)$ indicates the probability of w_t given previous sequence $\cdot$ in the backward or forward model respectively. So far, the model is able to generate a sentence containing an assigned word. But the target word is not used as a verb in the generated sequence all the time.

We regard a word with various parts-of-speech as a polyseme. If the word is used with a specific part-of-speech which we are concerned about, we label it with the specific part-of-speech tag, otherwise, the word remains unchanged. In our case, we use a POS tagger (Bird et al., 2009) to label all the words in the metaphorical pairs we extracted into two categories: verb and the other. For instance, "*spells*" is labeled as "*spells.v*" in the corpus if it is used as a verb otherwise "*spells*". We train the POS constrained model with the labeled corpus, and we generate sentences like "*she spells.v her husband at the wheel*" rather than "*he whispered spells as he moved his hands.*" when given "*spells.v*" as input.

3.3 Adjustable Joint Beam Search

In the end-to-end model, the goal of the decoder is to find a sequence $\hat{y}$ which maximizes the conditional probability given by a specific model θ and an input sequence x:

$$\hat{y} = argmax_{y\in\mathcal{Y}}p_\theta(y|x), \tag{2}$$

where $\mathcal{Y}$ is a set of all the possible sequences in the output space. It is impracticable to explore the whole space. Models decompose this problem into a sequence of time steps. The original beam search algorithm produces a probability distribution at each time step over the vocabulary V. And the log function is applied on the probability distribution to get the score distribution. Instead of simply choosing the token with the highest score, the beam search algorithm keeps top k candidates from a large matrix of dimensions $k \times |V|$ which expands the search space, where k denotes

[3] Keep in line with Mao et al. (2018)

[4] IN vectors are input vectors of a trained CBOW model. OUT vectors are output vectors of a trained CBOW model.

Algorithm 2 Adjustable Joint Beam Search

```
Require: $k$ : beam size
Require: $\alpha$, $\beta$: adjustment factors
Require: $L$ : maximum sequence length
Require: $w_1, w_2$ : input words
$Decoder - Init(w_i, n)$ copies the initial state of the decoder $n$ times when inputting $w_i$ to the decoder.
$Decoder - Step(beam)$ calculates the score distributions on the beam.
$Top - K(scores, b)$ selects $b$ candidates with highest scores in the score distribution and returns corresponding beam ids ($beam.id$) and word ids ($beam.indices$).
$t = 0$; $l = k$;$n = k/2$
$beam1 \leftarrow Decoder - Init(w_1, n)$
$beam2 \leftarrow Decoder - Init(w_2, n)$
while $l > 0$ and $t < L$ do
    $scores1 = Decoder - Step(beam1)$
    $scores2 = Decoder - Step(beam2)$
    $scores = \alpha \cdot scores1 + \beta \cdot scores2$
    $beam1 = Top - K(scores, n)$
    $beam2.ids \leftarrow beam1.ids + n$
    $beam2.indices \leftarrow beam1.indices$
    $t = t + 1$
    $n_e$ ←number of ("</s>") selected in $beam1$ and $beam2$.
    $l = l - n_e$
return $beam1, beam2$
```

the beam size. When "</s>" is selected among the highest scoring candidates the beam is reduced by one. When the beam is zero, the beam search algorithm stops (Lowerre, 1976; Post and Vilar, 2018).

As the metaphorical usage of the verb is represented by the metaphorical pair. We need to generate a sentence for the target verb where the contextual sense of the target word equals to the literal sense of its fit word, which means both target word and its fit word should be suitable in the generated sentence. Given two different verbs in a metaphorical pair as inputs (e.g. "*devoured*.v", "*enjoyed*.v"), we hope to generate a same context for them. However, the original beam search algorithm can hardly choose the same candidates at each time step for them. Yu et al. (2018) propose the joint beam search algorithm for pun generation. The algorithm selects candidates for the two inputs according to the joint score distribution on all beam while decoding. Nevertheless, the semantic distance between the metaphorical sense and the literal sense of the target verb is not the same with the distance between the metaphorical sense and contextual sense of the target word. In addition, the frequencies of the words in each metaphorical pair differ, and there is no reason to believe that two words in a metaphorical pair have the same influence on generating metaphors. Therefore, we use the adjustment factors α, β to modify the weights of the score distributions for two inputs. The proposed adjustable joint beam search algorithm is described in Algorithm 2.

Different from the original beam search algorithm, to start the adjustable joint beam search algorithm, the initial states of the decoder are copied $k/2$ times for each input. At each time step, candidates are chosen from a weighed summation of the score distributions from two sets of the beam. The beam for the next time step is filled by taking the states corresponding to the selected beam ids and word ids. Half of the chosen words are duplicate but come from distinct beam, which means the outputs are one-to-one correspondent for the two distinct inputs. Although the corresponding two sequences select the same word at each time step, they have distinct hidden states, and thus their score distributions are different. The decoder finally outputs k sentences in parallel and the corresponding two sentences are the same except for the input words. We find an intersection between a metaphorical usage and a literal usage of a verb by this means. To avoid that the generated sentence is semantically inclined to the word in a metaphorical pair whose frequency is relatively higher and take the second hypothesis (**H2**) into consideration, we calculate the adjustment factors as follows:

$$\alpha = \sigma(1 - \frac{wf(word1)}{wf(word1) + wf(word2)}), \quad (3)$$

$$\beta = \sigma(1 - \frac{wf(word2)}{wf(word1) + wf(word2)}), \quad (4)$$

where $wf(a)$ denotes the frequency of the word a in the corpus. As metaphorical senses of words occur with relatively lower frequency in the corpus, we adjust the weights negatively correlated to the word frequency. And we use σ, the sigmoid function which is differentiable and widely used in the neural network models, to smooth the adjustment factors.

4 Experiments

4.1 Data Preprocessing

Adequate data is a prerequisite condition for training a good language model. It is unrealistic to achieve the goal based on the limited annotated metaphor corpus. In this paper, we use the English Wikipedia corpus to train the POS constrained language model. The corpus is split into sentences whose maximum length is 50 words. We lowercase and tokenize the sentences. All the numeric tokens are replaced with "#". We automatically extract 2812 metaphorical pairs from the corpus, and label words in them into two forms: "*word.v*" and "*word*". We use 461,685 sentences as the training set. We keep 120,000 most frequently occurring words and replace other words with the "<*unk*>" token. We call this processed corpus a **normal corpus**. For comparison, we extract the metaphors in the normal corpus using an unsupervised metaphor identification approach (Mao et al., 2018) and build a **metaphor corpus** with 310,908 sentences. We keep the same vocabulary size. To explore the influence of a fine-grained sense in the metaphor generation task, we use a Word Sense Disambiguation (WSD) tool[5] to label the verbs in the normal corpus. As the polysemes are regarded as different words which obviously increases the vocabulary size, we keep 165,000 most frequently occurring words. And this corpus is named **sense corpus**.

4.2 Compared Models

Since there has been no neural model applied on metaphor generation and the previous template-based models can not generate such verb-oriented metaphors, we implement six neural models for comparison and explore the intrinsic characteristics of metaphor generation. Models are trained on the **normal corpus**, unless otherwise specified.

Normal Model: This model is a basic end-to-end framework whose inputs are verbs and outputs are sentences. We use the teacher forcing algorithm (Williams and Zipser, 1989) while training. The input of the encoder is a verb, and the reference for the decoder is a sentence containing the verb.

POS constrained Model: This model combines the basic end-to-end framework with the POS constrained language model. The model can generate a sentence containing a given verb.

Fit Word Model: The model is trained in the same way as the POS constrained model. The input is a fit word, and we directly replace the fit word in the generated sentence with the corresponding target word.

Metaphor Based Model: The model is the same as the POS constrained model but trained on the **metaphor corpus**.

Uncommon Sense Model: Based on the second hypothesis(**H2**), metaphorical sense appears less common in the corpus than literal sense. It is intuitive to associate the metaphorical sense of the target word with a low-frequency sense. We keep the senses of a target word which appear in the corpus more than 9 times in a sense list. Then we choose the sense with the lowest frequency in the sense list as the metaphorical sense of the target verb. This model generates sentences similarly to the POS constrained model, except that it is trained on the **sense corpus.**

Adjustable Joint Model: The training process is exactly the same as the POS constrained model's, but we use the adjustable joint beam search algorithm while decoding.

4.3 Automatic Evaluation

We extract the metaphorical pairs automatically in the normal corpus and conduct experiments. The automatic evaluation results are shown in Table 1.

Model	l.ave	w.num	ppl.
Normal	10.34	16029	97.84
POS constrained	10.51	16361	133.24
Fit Word	10.70	16666	154.49
Metaphor Based	9.37	14526	**61.88**
Uncommon Sense	7.45	11569	96.52
Adjustable Joint	**11.35**	**16887**	97.03

Table 1: Automatic evaluation results for generated metaphors based on the automatically extracted verbs.

Each target verb may form distinct metaphorical pairs with different fit words. Both the normal model and the POS constrained model generate the same sentences for the same target verbs while the fit word model generates the same sentences for the same fit words. Taking the metaphorical pairs into consideration, adjustable joint model can generate various sentences when the target words are the same but used in diffe-

[5] https://github.com/alvations/pywsd

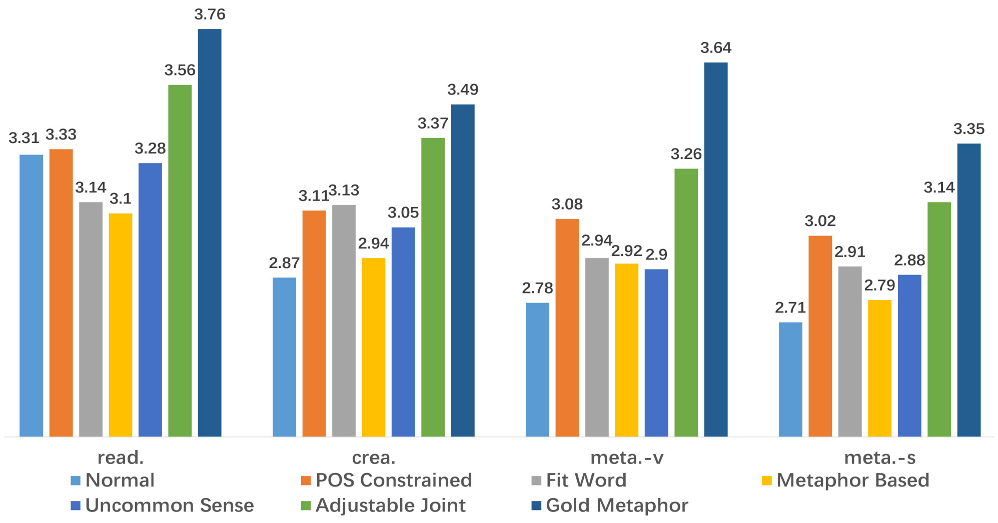

Figure 3: Results of human evaluation.

rent senses. For a fair quantitative analysis, each model generates 1555 sentences for distinct target verbs. **Length-average (l.ave)** is the average length of the generated sentences of each model. **Word.number(w.num)** is the total number of the distinct words in the generated sentences, and the adjustable joint model decodes words with the highest diversity compared to the other models. We use the language modeling toolkit SRILM[6] to evaluate the **perplexity scores (ppl.)**. Theoretically, the normal model should generate more fluent sentences without considering the constraint that a given word must appear in the outputs, however the ppl. is calculated as Eq 5 shows:

$$ppl = 10\hat{}\left(\frac{-logP(T)}{s.num + w.num - OOVs}\right), \quad (5)$$

where $P(T)$ denotes the probabilities of all the sentences, $s.num$ denotes the total number of the sentences. $OOVs$ denotes the number of out-of-vocabulary words. The outputs of the metaphor based models tend to be shorter which results in a higher $P(T)$ and thus a low ppl. Although the adjustable joint model generates sentences with lexical constraints, it chooses the candidates with the highest joint score at each time step and uses more distinct words, and thus it obtains a high $P(T)$ as well as a large $w.num$. It achieves a relatively low ppl. What's more, all the sentences generated by the adjustable joint model are novel and do not exist in the corpus according to the duplicate checking . The fit word model decodes according to the fit words and then the fit words in the sentences are replaced by their target words directly, which results in a higher ppl. Since the amount of the training data corresponding to the inputs of the uncommon sense model is the least, the generated sentences are not so fluent.

4.4 Human Evaluation

For a thorough comparison, we select 80 gold metaphors with high confidence (>0.6) from the data set proposed by (Mohammad et al., 2016). Each verb in the data set was annotated for metaphoricity by 10 annotators and we use the verbs in the selected metaphors as the target words. As metaphor is such a creative and delicate language that automatic evaluation is not adequate. We ask native English speakers on Amazon Mechanical Turk to evaluate all the sentences generated by the neural models and corresponding gold metaphors in four aspects. Each sentence is scored from 1 to 5 by 5 judges. **Readability(read.)** indicates whether the sentence is fluent and consistent with the rules of grammar. **Creativity(crea.)** indicates whether the sentence is distinct and creative. **Metaphorical or Literal Usage of the Verb(meta.-v)** indicates whether the target word is used literally or metaphorically. 1 denotes that the usage of the verb is definitely literal and 5 denotes the verb is obviously metaphorically used. We display typical properties of metaphorical and literal senses as follows: literal usages tend to be more basic, straightforward meaning, more physical and closely tied to our senses; Metaphorical usages tend to be more abstract, more vague and often surprising, someti-

[6] http://www.speech.sri.com/projects/srilm/

Model	Generated Sentences
Target Verb: *absorbed*	Fit Word: *learn*
Normal	*it is absorbed by the united states .*
POS constrained	*it absorbed the united states in the early century .*
Fit Word	*they absorbed that they are able to find themselves .*
Metaphor Based	*they absorb water from the water to be used as a result of the disease .*
Uncommon Sense	*he absorbed more than a few hundred feet .*
Adjustable Joint	***he absorbed his studies at the university of birmingham .***
Gold Metaphor	*he absorbed the knowledge or beliefs of his tribe .*
Target Verb: *pour*	Fit Word: *crowd*
Normal	*while drinking , he is able to kill him .*
POS constrained	*being poured , the band was released by the band .*
Fit Word	*they poured for the first time , and the team was the first to win the championship .*
Metaphor Based	*it was poured in # and # .*
Uncommon Sense	*she poured the police and her husband .*
Adjustable Joint	***millions of trees poured out of the area .***
Gold Metaphor	*People were pouring out of the theater.*

Table 2: Examples of generated sentences.

mes bring in imagery from a different domain. The aspects above are rated according to comparison. **Metaphoricity of the sentence(meta.-s)** indicates whether the sentence is a metaphor. The score ratings are defined as: 1. not a metaphor at all; 2. pathetic metaphor; 3. not-so-bad metaphor; 4. interesting metaphor; 5. gold metaphor.

Results are shown in Figure 3. The metaphor based model generates shorter sentences which results in a low **ppl.**, but it replicates some words at times, which makes it difficult for human to interpret. The fit word model replaces the fit words with the target words directly and may break the grammatical collocations. Although the adjustable joint model ensures the intersection sense of two words in the metaphorical pair to appear in one sentence, the generated utterance is still readable. As for the **crea.**, without any lexical constraint, the normal model always generates the sentences which are similar to the relatively high-frequency sentences in the corpus and results in the lack of novelty. In contrast, the target words are used in its less used literal senses in the uncommon sense model and the results seem to be kind of creative. The adjustable joint model performs closely to the gold metaphor on the **crea.**, which proves its ability in modeling the creative usage of the verbs. **meta.-v** directly reveals the capability of modeling the metaphorical senses of the target verb. The normal model is uncompetitive as it can not even ensure the appearance of the target verb. The difference between the POS constrained model and the uncommon sense model inspires us that although the metaphorical senses of the words occur with relatively lower frequency in the corpus than their literal senses, the metaphorical senses cannot be easily defined as the most uncommon senses of the target verbs, and thus the exploration on modeling the metaphorical senses is essential. As for the **meta.-s**, it reveals the comprehensiveness of the sentence-level metaphor generation and reminds us of modeling the metaphors in a more thoughtful perspective. The adjustable joint model outperforms the other models on the 3 aspects (**crea.**, **meta.-v**, **meta.-s**) significantly with paired t-test with $p < 0.05$.

4.5 Case Study

To illustrate concretely, we show some examples generated by different models in Table 2. Both the normal model and the POS constrained model generate sentences without considering the senses of the verbs and tend to generate monotonous sentences. And the fit word model generates sentences which are suitable for the fit word but inappropriate for the target word. For example, "*they crowed for the first time*" is fluent while "*they poured for the first time*" is strange. Other models decoding with a sense constraint generate sentences conveying the assigned senses to different degree. However, the uncommon sense model is fine-grained and the training corpus is annotated with WSD tools which results in not only the lack of corpora for some senses but also a sense-label error, as there are no WSD tools that could tag the senses of the verbs with a high precision (> 0.6) (Luo et al., 2018b,a). To solve these problems, we use the extracted metaphorical pairs to depict a metaphorical sense and the generated metaphors are readable.

We also explore the shared inferential structure of a metaphorical usage and a literal usage of a verb by changing the adjustable factors. Table 2 demonstrates the semantic shift of the verb. The POS constrained model can be seen as a special adjustable joint model whose adjustable factors are $\alpha = 1, \beta = 0$ and it generates sentences only considering the target words. In this way the target words are literally used. For contrast, the fit word model with a special setting of the adjustable factors $\alpha = 0, \beta = 1$ generates sentences completely dependent on the fit words, which may result in sentences that are not semantically appropriate. When α, β are caculated by Eq 3 and Eq 4, the generated sentences covey the metaphorical senses of verbs.

5 Conclusion

We make an exploration on verb-oriented metaphor generation and propose a neural approach on automatic metaphor generation. The approach identifies metaphorically used verbs in the normal corpus and extracts the metaphorical pairs in the sentences. We propose a POS constrained model to ensure the appearance of the given verbs and decode with the adjustable joint beam search algorithm, which takes the metaphorical senses of the given verbs into consideration. We generate metaphors which are not only fluent and readable but also creative. However, we can only generate metaphors based on the verbs whose metaphorical senses can be found in the corpus. For future work, we will explore techniques to generate metaphors without extracting its fit words in the corpus and improve the quality of generated metaphors.

Acknowledgment

This work was supported by National Natural Science Foundation of China (61772036) and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology). We appreciate Hongyu Zang for the detailed suggestions and thoughts. We thank Fuli Luo, Pengcheng Yang, Rui Mao and the anonymous reviewers for their helpful comments. Xiaojun Wan is the corresponding author.

References

Keiga Abe, Kayo Sakamoto, and Masanori Nakagawa. 2006. A computational model of the metaphor generation process. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 28.

Rodrigo Agerri, John Barnden, Mark Lee, and Alan Wallington. 2007. Metaphor, inference and domain independent mappings. In *Proceedings of RANLP*, pages 17–23. Citeseer.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. Ö'Reilly Media, Inc.".

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

Jerome Feldman and Srinivas Narayanan. 2004. Embodied meaning in a neural theory of language. *Brain and language*, 89(2):385–392.

Lakoff G. and Johnson M. 1985. Metaphors we live by. *Artif. Intell.*, 27(3):357–361.

Michael R. Genesereth. 1980. Metaphors and models. In *Proceedings of the 1st Annual National Conference on Artificial Intelligence, Stanford University, CA, USA, August 18-21, 1980.*, pages 208–211.

Raquel Hervás, Rui P Costa, Hugo Costa, Pablo Gervás, and Francisco C Pereira. 2007. Enrichment of automatically generated texts using metaphor. In *Mexican International Conference on Artificial Intelligence*, pages 944–954. Springer.

Mark Alan Jones. 1992. Generating a specific class of metaphors. In *Proceedings of the 30th annual meeting on Association for Computational Linguistics*, pages 321–323. Association for Computational Linguistics.

Jenny Lederer. 2016. Finding metaphorical triggers through source (not target) domain lexicalization patterns. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 1–9.

Linlin Li and Caroline Sporleder. 2010. Using gaussian mixture models to detect figurative language in context. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 2-4, 2010, Los Angeles, California, USA*, pages 297–300.

Bruce T Lowerre. 1976. The harpy speech recognition system. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF COMPUTER SCIENCE.

Fuli Luo, Tianyu Liu, Zexue He, Qiaolin Xia, Zhifang Sui, and Baobao Chang. 2018a. Leveraging gloss knowledge in neural word sense disambiguation by hierarchical co-attention. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1402–1411.

Fuli Luo, Tianyu Liu, Qiaolin Xia, Baobao Chang, and Zhifang Sui. 2018b. Incorporating glosses into neural word sense disambiguation. *arXiv preprint arXiv:1805.08028*.

Rui Mao, Chenghua Lin, and Frank Guerin. 2018. Word embedding and wordnet based metaphor identification and interpretation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics (ACL).

James H Martin. 1990. *A computational model of metaphor interpretation*. Academic Press Professional, Inc.

James H Martin. 2006. A corpus-based analysis of context effects on metaphor comprehension. *TRENDS IN LINGUISTICS STUDIES AND MONOGRAPHS*, 171:214.

Zachary J. Mason. 2004. Cormet: A computational, corpus-based conventional metaphor extraction system. *Computational Linguistics*, 30(1):23–44.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

George Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Saif Mohammad, Ekaterina Shutova, and Peter Turney. 2016. Metaphor as a medium for emotion: An empirical study. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 23–33.

Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. 2016. Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation. *arXiv preprint arXiv:1607.00970*.

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*, pages 83–84. International World Wide Web Conferences Steering Committee.

Srini Narayanan. 1997. Knowledge-based action representations for metaphor and aspect (karma). *Computer Science Division, University of California at Berkeley dissertation*.

Geoffrey Nunberg. 1987. Poetic and prosaic metaphors. *Theoretical Issues in Natural Language Processing 3*.

Ekaterina Ovchinnikova, Vladimir Zaytsev, Suzanne Wertheim, and Ross Israel. 2014. Generating conceptual metaphors from proposition stores. *arXiv preprint arXiv:1409.7619*.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324. Association for Computational Linguistics.

Marek Rei, Luana Bulat, Douwe Kiela, and Ekaterina Shutova. 2017. Grasping the finer point: A supervised similarity network for metaphor detection. *arXiv preprint arXiv:1709.00575*.

Ekaterina Shutova. 2010. Models of metaphor in NLP. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden*, pages 688–697.

Ekaterina Shutova, Douwe Kiela, and Jean Maillard. 2016. Black holes and white rabbits: Metaphor identification with visual features. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 160–170.

Gerard Steen. 2010. *A method for linguistic metaphor identification: From MIP to MIPVU*, volume 14. John Benjamins Publishing.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1171–1181.

Asuka Terai and Masanori Nakagawa. 2010. A computational system of metaphor generation with evaluation mechanism. In *International Conference on Artificial Neural Networks*, pages 142–147. Springer.

Tony Veale. 2016. Round up the usual suspects: Knowledge-based metaphor generation. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 34–41.

Yorick Wilks. 1978. Making preferences more active. *Artif. Intell.*, 11(3):197–223.

Ronald J Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. SGM: sequence generation model for multi-label classification. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 3915–3926.

Zhiwei Yu, Jiwei Tan, and Xiaojun Wan. 2018. A neural approach to pun generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1650–1660.

Hongyu Zang and Xiaojun Wan. 2017. Towards automatic generation of product reviews from aspect-sentiment scores. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 168–177.