A Two Step Process for Generating Sentences from the Sums of their Embeddings

Abstract

Converting a sentence to a meaningful vector representation has uses in many NLP tasks, however very few methods allow that representation to be restored to a human readable sentence. Being able to generate sentences from the vector representations is expected to open up many new applications. We introduce such a method for moving from sum of word embedding representations back to the original sentences. This is done using a two stage process; first a greedy algorithm is utilised to convert the vector to a bag of words, which is then ordered using a simple probabilistic language model to get back the sentence. To our knowledge this is the first work to demonstrate qualitatively the ability to reproduce text from a large corpus based on its sentence embeddings. As well as practical applications for sentence generation, the success of this method prompts further theoretical investigations the degree of information maintained by the sum of embeddings representation.

1 Introduction

We present a method for generating sentences based on vector representations of the sum of their word embeddings. The generation task, going from any vector representation back to a sentence, is quite challenging. It has not received a lot of attention.

Dinu and Baroni (2014) motivates this work from a theoretical perspective given that a sentence encodes its meaning, and the vector encodes the same meaning, then it must be possible to translate in both directions between the natural language and the vector representation. An implementation, such as the work reported in this paper, which demonstrates the truth of this dual space theory, has its own value. There are also many potential practical applications of such an implementation, often ranging around certain types of "translation" tasks.

A number of techniques for learning to associate various media and sentences to a common vector space have been proposed. Such as Farhadi et al. (2010) and Socher et al. (2014) for images; Kågebäck et al. (2014) and Yogatama et al. (2015) for multi-document summaries, and Zhang et al. (2014) for sentences in multiple languages. However, all of these techniques are tied to being able to use the vector space to compare the sentences and other media for similarities. With appropriate generative models for sentence vectors, these "matching" based solutions – which find the most similar sentence to a vector from a list; become "generative" solutions – which generate a new sentence entirely.

A sentence generation method has as its input a sentence embedding, and outputs the sentence which it corresponds to as. The input is a vector, for example $\tilde{s} = [0.11, 0.57, -0.21, ..., 1.29]$, and the output is a sentence, for example "The boy was happy.". That input vector could come direct as the sentence embedding representation of a reference sentence (as is the case for the evaluation presented here). More practically it could come as the output of some other process; for example a machine learnt mapping from an image to the vector representation of its textual description. This vector representation is transformed through the process into a human readable sentence, describing for example the image.

The current state of the art for sentence generation do produce human readable sentences, but they are only rough approximation to the intended sentence. These existing works are those of Iyyer et al. (2014)

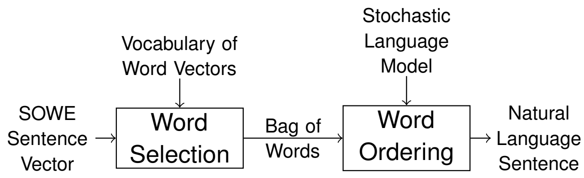

Figure 1: The Sel. BOW+Ord. process for the regenerating sentences from SOWE-type sentence vectors.

and Bowman et al. (2015). Both these have been demonstrated to produce full sentences. These sentences are qualitatively shown to be loosely similar in meaning to the original sentences. Neither work has produced quantitative evaluation, making it hard to compare their performance. Both are detailed further in Section 2.

Our method performs the sentence generation in two steps, as shown in Figure 1. It combines the work of White et al. (2016) on generating bags of words (BOW) from SOWE (Word Selection); with the work of Horvat and Byrne (2014) on ordering BOW into sentences (Word Ordering). The overall two step approach can generate proper sentences from SOWE vectors.

The rest of the paper is organized into the following sections. Section 2 introduces the area, discussing in general sentence models, and prior work on generation. Section 3 explains the problem in detail and how the two step method is used to solving it. Section 4 describes the settings used for evaluation. Section 5 presents the results on this evaluation. The paper concludes with Section 6 and a discussion of future work on this problem.

2 Related Works

2.1 Embedding Models

Ritter et al. (2015) and White et al. (2015) found that when classifying sentences into categories according to meaning, simple SOWE outperformed more complex models. Both works used sentence embeddings as the input to classifiers. Ritter et al. (2015) classified challenging artificial sentences into categories based on the positional relationship described using Naïve Bayes. White et al. (2015) classified real-world sentences into groups of semantically equivalent paraphrases. In both cases, they found the best SOWE-type sentence embeddings to be amongst the highest performing representations. In the case of Ritter et al. (2015) it outperformed the next best representation by over 5%. In the case of White et al. (2015) it was within a margin of 1% from the very best performing method. These results suggest there is high consistency in the relationship between a point in the SOWE space, and the meaning of the sentence. Thus this simple method is worth further consideration. SOWE is the basis of the work presented in this paper.

2.2 Sentence Generation from Vector Embeddings

To the best of our knowledge only three prior works exist in the area of sentence generation from embeddings. The first two (Dinu and Baroni (2014), Iyyer et al. (2014)) of which are based on compositional embeddings, while the most recent work at the time of this writing (Bowman et al. (2015)), is however a non-compositional approach.

Dinu and Baroni (2014) extends the models described by Zanzotto et al. (2010) and Guevara (2010) for generation. The composition is described as a linear transformation of the input word embeddings to get an output vector, and another linear transformation to reverse the composition reconstructing the input. The linear transformation matrices are solved for using least squares regression. This method of composing, can be applied recursively from words to phrases to clauses and so forth. It theoretically generalises to whole sentences, by recursively applying the composition or decomposition functions. However, Dinu and Baroni's work is quantitatively assessed only on direct reconstruction for decomposing Preposition-Noun and Adjective-Noun word phrases. In these cases where the decomposition function was trained on directly vectors generated using the duel composition function they were able to get perfect reconstruction on the word embedding based inputs.

Iyyer et al. (2014) extends the work of Socher et al. (2011) defining an unfolding recursive dependency-tree recursive autoencoder (DT-RAE). Recursive neural networks are jointly trained for both composing the sentence's words into a vector, and for decomposing that vector into words. This composition and decomposition is done by reusing a composition neural network at each vertex pf the dependency tree structure, with different weight matrices for each dependency relation. The total network is trained based on the accuracy pf reproducing its input word embeddings. It can be used to generate sentences, if a dependency tree structure for the output is provided. This method was demonstrated quantitatively on five examples (shown in Table 3); the generated sentences were shown to be loosely semantically similar to the originals.

Bowman et al. (2015) uses a a modification of the variational autoencoder (VAE) (Kingma and Welling, 2013) with natural language inputs and outputs, to learn the sentence representations. These input and output stages are performed using long short-term memory recurrent neural networks (Hochreiter and Schmidhuber, 1997). They demonstrate a number of uses of this technique, one of which is sentence generation, in the sense of this paper. While being a generative model it does not seek to recreate a sentence purely from its vector input, but rather to produce a series of probability distributions on the words in the sentence. These distributions can be evaluated greedily, which the authors used to give three quantitative examples of resynthesis (also shown in Table 4). They found the sentence embeddings created captured largely syntactic and loose topical information.

None of the existing methods have demonstrated recreation of a full sentence input close enough to allow for quantitative evaluation on a full corpus. They tend to output lose paraphrases, or roughly similar sentences – itself a separately useful achievement. That is not the case for our method described in the next section, which can often exactly recreate the original sentence from its vector representation.

Unlike current sentence generation methods, the non-compositional BOW generation method of White et al. (2016) generally outputs a BOW very close to the reference for that sentence – albeit at the cost of losing all word order information. It is because of this accuracy that we base our proposed sentence generation method on it (as detailed in Section 3.1). The Word Selection step we used is directly based on their greedy BOW generation method. We improve it for sentence generation by composing with a word ordering step to create the two step sentence generation process.

3 General Framework

As discussed in Section 1, and shown in Figure 1, the approach taken to generate the sentences from the vectors comes in two steps. First selecting the words used – this is done deterministically, based on a search of the embedding space. Second is to order them, which we solve by finding the most likely sequence according to a stochastic language model. The separation of the process into two steps is unlike any of the existing methods for sentence generation from vectors. The two subproblems which result from this split resemble more classical NP-Hard computer science problems; thus variations on known techniques can be used to solve them.

3.1 Word Selection

White et al. (2016) approaches the BOW generation problem, by as a the vector selection problem – selecting the vectors that sum up closest to a given vector. This is related to the knapsack and subset sum problems. They formally define the vector selection problem as:

$$(\tilde{s}, \mathcal{V}, d) \mapsto \underset{\left\{\forall \tilde{c} \in \mathbb{N}_0^{|\mathcal{V}|}\right\}}{\operatorname{argmin}} \; d(\tilde{s}, \sum_{\tilde{x}_j \in \mathcal{V}} \tilde{x}_j c_j)$$

to find the bag of vectors selected from the vocabulary set $\mathcal{V}$ which when summed is closest to the target vector $\tilde{s}$. Closeness is assessed with distance metric d. $\tilde{c}$ is the indicator function for that multi-set of vectors. As there is a one to one correspondence between word embeddings and their words, finding the vectors results in finding the words. White et al. (2016) propose a greedy solution to the problem.

The key algorithm proposed by White et al. (2016) is greedy addition. The idea is to greedy add vectors to a partial solution building towards a complete bag. This starts with an empty bag of word embeddings, and at each step the embedding space is searched for the vector which when added to the current partial solution results in the minimal distance to the target – when compared to other vectors from the vocabulary. This step is repeated until there are no vectors in the vocabulary that can be added without moving away from the solution. Then a fine-tuning step, n-substitution, is used to remove some simpler greedy mistakes.

The n-substitution step examines partial solutions (bags of vectors) and evaluates if it is possible to find a better solution by removing n elements and replacing them with up-to n different elements. The replacement search is exhaustive over the n-ary cartesian product of the vocabulary. Only for $n = 1$ is it currently feasible for practical implementation outside of highly restricted vocabularies. Never-the-less even 1-substitution can be seen as lessening the greed of the algorithm, through allowing early decisions to be reconsidered in the full context of the partial solution. The algorithm does remain greedy, but many simple mistakes are avoided by n-substitution. The greedy addition and n-substitution processes are repeated until the solution converges.

3.2 The Ordering Problem

After the bag of words has been generated by the previous step, it must be ordered. For example "are how , today hello ? you", is to be ordered into the sentence: "hello , how are you today ?". This problem cannot always be solved to a single correct solution. Mitchell and Lapata (2008) gives the example of "It was not the sales manager who hit the bottle that day, but the office worker with the serious drinking problem." which has the same word content (though not punctuation) as "That day the office manager, who was drinking, hit the problem sales worker with a bottle, but it was not serious.". However, while a unique ordering cannot be guaranteed, finding the most likely word ordering is possible.

Horvat and Byrne (2014) formulated the word ordering problem as a generalised asymmetrical travelling salesman problem (GA-TSP). Figure 2 shows an example of the connected graph for ordering five words. We extend beyond the approach of Horvat and Byrne (2014) by reformulated the problem as a linear mixed integer programming problem (MIP). This allows us to take advantage of the efficient existing solvers for this problem. Beyond the GA-TSP approach, direct MIP formulation allows for increased descriptive flexibility and opens the way for further enhancement. The description is freed of some constraints of a TSP. For example, word ordering does have distinct and known start and end nodes (as shall be detailed in the next section). To formulate it as a GA-TSP it must be a tour without beginning or end. Horvat and Byrne (2014) solve this by simply connecting the start to the end with a zero cost link. This is not needed if formulating this as a MIP problem, the start and end nodes can be treated as special cases. Being able to special case them as nodes known always to occur allows some simplification in the subtour elimination step. The formulation to mixed integer programming is otherwise reasonably standard.

3.2.1 Notation

We will write w_i to represent a word from the bag $\mathcal{W}$ ($w_i \in \mathcal{W}$), with arbitrarily assigned unique subscripts. Where a word occurs with multiplicity greater than 1, it is assigned multiple subscripts, and is henceforth treated as a distinct word.

Each vertex is a sequence of two words, $\langle w_i, w_j \rangle \in \mathcal{W}^2$. This is a Markov state, consisting of a word w_j and its predecessor word w_i – a bigram.

Each edge between two vertices represents a transition from one state to another which forms a trigram. The start vertex is given by $\langle w_\blacktriangleright, w_\triangleright \rangle$, and the end by $\langle w_\triangleleft, w_\blacktriangleleft \rangle$. The pseudowords $w_\blacktriangleright, w_\triangleright, w_\triangleleft, w_\blacktriangleleft$ are added during the trigram models training allowing knowledge about the beginning and ending of sentences to be incorporated.

The GA-TSP districts are given by the sets of all states that have a given word in the first position. The district for word w_i is given by $S(w_i) \subseteq \mathcal{W}^2$, defined as $S(w_i) = \{\langle w_i, w_j \rangle \mid \forall w_j \in \mathcal{W}\}$. It is required to visit every district, thus it is required to use every word. With this description, the problem can be formulated as a MIP optimisation problem.

3.2.2 Optimization Model

Every MIP problem has set of variables to optimise, and a cost function that assesses how optimal a given choice of values for that variable is. The cost function for the word ordering problem must represent how unlikely a particular order is. The variables must represent the order taken. The variable are considered as a table which indicates if a particular transition between states is taken. Note that for any pair of Markov states $\langle w_a, w_b \rangle, \langle w_c, w_d \rangle$ is legal if and only if $b = c$, so we denote legal transitions as $\langle w_i, w_j \rangle \to \langle w_j, w_k \rangle$. Such a transition has cost:

$$C[\langle w_i, w_j \rangle, \langle w_j, w_k \rangle] = -\log\left(P(w_k | w_i, w_j)\right)$$

The table of transitions to be optimized is:

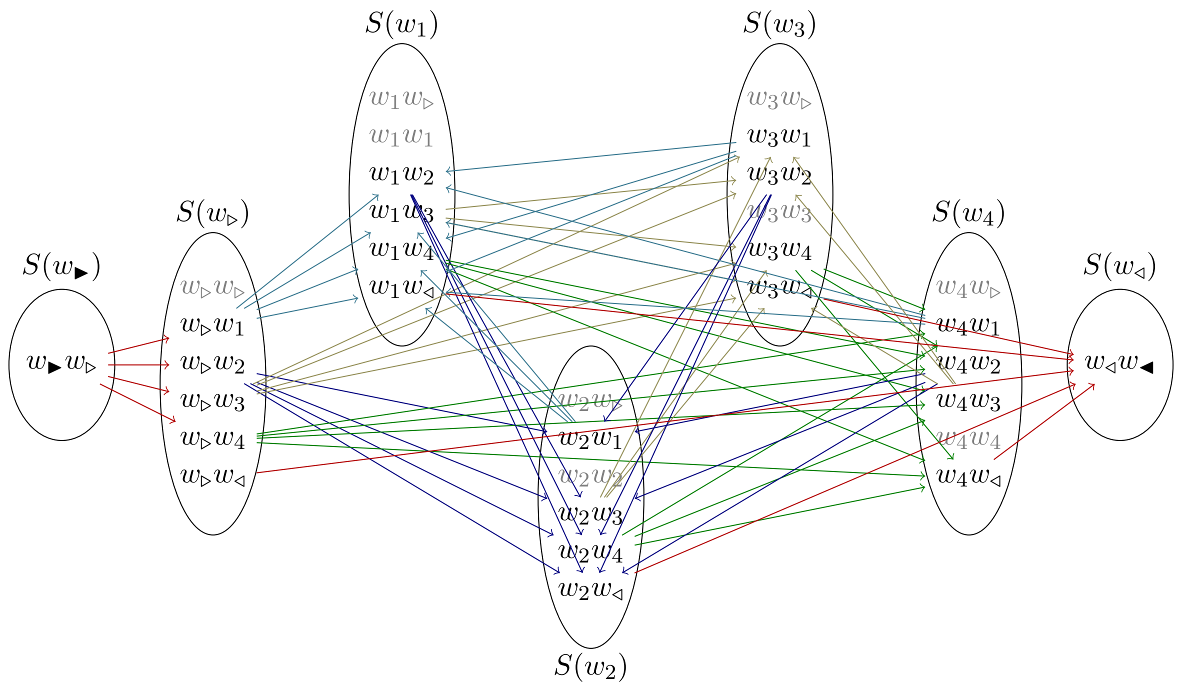

Figure 2: A graph showing the legal transitions between states, when the word-ordering problem is expressed similar to a GA-TSP. Each edge $(w_a w_b) \to (w_b w_c)$ has cost $-\log(P(w_c \mid w_a w_b)$. The nodes are grouped into districts (word). Nodes for invalid states are greyed out.

$$\tau[\langle w_i, w_j\rangle,\ \langle w_j, w_k\rangle] = \begin{cases} 1 & \text{if transition from} \\ & \langle w_i, w_j\rangle \to \langle w_j, w_k\rangle \text{ occurs} \\ 0 & \text{otherwise} \end{cases}$$

The total cost to be minimized, is given by

$$C_{total}(\tau) = \sum_{\forall w_i, w_j, w_k \in \mathcal{W}^3} \tau[\langle w_i, w_j\rangle,\ \langle w_j, w_k\rangle] \cdot C[\langle w_i, w_j\rangle,\ \langle w_j, w_k\rangle]$$

The probability of a particular path (i.e. of a particular ordering) is thus given by $P(\tau) = e^{-C_{total}(\tau)}$

The word order can be found by following the links. The function $f_\tau(n)$ gives the word that, according to τ occurs in the nth position.

$$\begin{aligned} f_\tau(1) =& \{w_a \mid \tau[\langle w_\blacktriangleright, w_\triangleright\rangle, \langle w_\triangleright, w_a\rangle] = 1\}_1 \\ f_\tau(2) =& \{w_b \mid \tau[\langle w_\triangleright, f_\tau(1)\rangle, \langle f_\tau(1), w_b\rangle] = 1\}_1 \\ f_\tau(n) =& \\ & \underset{\text{when} \quad n \geq 3}{\{w_c \mid \tau[\langle f_\tau(n-2), f_\tau(n-1)\rangle, \langle f_\tau(n-1), w_c\rangle] = 1\}_1} \end{aligned}$$

The notation $\{\cdot\}_1$ indicates taking a singleton set's only element. The constraints on τ ensure that each set is a singleton.

3.2.3 Constraints

The requirements of the problem, place various constraints on to τ: The Markov state must be maintained: $\forall \langle w_a, w_b\rangle, \langle w_c, w_b\rangle \in \mathcal{W}^2$:

$$w_b \neq w_c \implies \tau[\langle w_a, w_b\rangle,\ \langle w_c, w_d\rangle] = 0$$

Every node entered must also be exited – except those at the beginning and end.
$\forall \langle w_i, w_j\rangle \in \mathcal{W}^2 \backslash \{\langle w_\blacktriangleright, w_\triangleright\rangle, \langle w_\triangleleft, w_\blacktriangleleft\rangle\}$:

$$\sum_{\forall \langle w_a, w_b\rangle \in \mathcal{W}^2} \tau[\langle w_a, w_b\rangle, \langle w_i, w_j\rangle] = \sum_{\forall \langle w_c, w_d\rangle \in \mathcal{W}^2} \tau[\langle w_i, w_j\rangle, \langle w_c, w_d\rangle]$$

Visit (enter) every district exactly once. i.e. use every word exactly once. $\forall w_i \in \mathcal{W} \backslash \{w_\blacktriangleright, w_\blacktriangleleft\}$:

$$\sum_{\forall \langle w_i, w_j\rangle \in S(w_i)} \sum_{\forall \langle w_a, w_b\rangle \in \mathcal{W}^2} \tau[\langle w_a, w_b\rangle,\ \langle w_i, w_j\rangle] = 1$$

To allow the feasibility checker to detect if ordering the words is impossible, transitions of zero probability are also forbidden. i.e. if $P(w_n | w_{n-2}, w_{n-1}) = 0$ then $\tau[\langle w_{n-2}, w_{n-1}\rangle, \langle w_{n-1}, w_n\rangle] = 0$. These transitions, if not expressly forbidden, would never occur in an optimal solution in any case, as they have infinitely high cost.

Lazy Subtour Elimination Constraints The problem as formulated above can be input into a MIPS solver. However, like similar formulations of the travelling salesman problem, some solutions will have subtours. We take the usual method for handling this; we use callbacks to impose lazy constraints to forbid such solutions at run-time. However, the actual formulation of those constraints are different to a typical GA-TSP.

Given a potential solution τ meeting all other constraints:

The core path – which starts at $\langle w_{\blacktriangleright}, w_{\triangleright}\rangle$ and ends at $\langle w_{\triangleleft}, w_{\blacktriangleleft}\rangle$ can be found. This is done by practically following the links from the start node, and accumulating them into a set $T \subseteq \mathcal{W}^2$

From the core path, the set of words covered is given by $\mathcal{W}_T = \{w_i \mid \forall\langle w_i, w_j\rangle \in T\} \cup \{w_{\blacktriangleleft}\}$. If $\mathcal{W}_T = \mathcal{W}$ then there are no subtours and the core-path is the complete path. Otherwise, there is a subtour to be eliminated.

If there is a subtour, then a constraint must be added to eliminate it. The constraint we define is that there must be a connection from at least one of the nodes in the district covered by the core path to one of the nodes in the districts not covered.

The districts covered by the tour are given by $S_T = \bigcup_{w_t \in \mathcal{W}_T} S(w_t)$. The subtour elimination constraint is given by

$$\sum_{\forall\langle w_{t1}, w_{t2}\rangle \in S_T} \sum_{\forall\langle w_a, w_b\rangle \in \mathcal{W}^2 \setminus S_T} \tau[\langle w_{t1}, w_{t2}\rangle, \langle w_a, w_b\rangle] \geq 1$$

i.e. there must be a transition from one of the states featuring a word that is in the core path, to one of the states featuring a word not covered by the core path.

This formulation around the notion of a core-path that differs this from typical subtour elimination in a GA-TSP. GA-TSP problems are not generally guaranteed to have any nodes which must occur. However every word ordering problem is guaranteed to have such a node – the start and end nodes. Being able to identify the core path allows for reasonably simple subtour elimination constraint definition. Other subtour elimination constraints do exist.

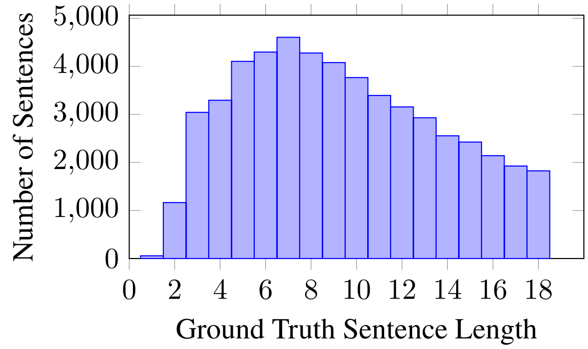

Figure 3: The distribution of the evaluation corpus after preprocessing.

4 Experimental Setup and Evaluations

This experimental data used in this evaluation was obtained from the data released with White et al. (2016).

4.1 Word Embeddings

GloVe representations of words are used in our evaluations (Pennington et al., 2014). There are many varieties of word embeddings which function with our algorithm. GloVe was chosen become of the availability of a large pre-trained vocabulary of vectors. The representations used for evaluation were pretrained on 2014 Wikipedia and Gigaword 5[1]. Other vector representations are presumed to function similarly.

4.2 Corpus and Language Modelling

The evaluation was performed on a subset of the Books Corpus (Zhu et al., 2015). The corpus was preprocessed as in the work of White et al. (2016). This meant removing any sentences which used words not found in the embedding vocabulary.

After preprocessing, the base corpus, was split 90:10. 90% (59,694,016 sentences) of the corpus was used to fit a trigram model. This trigram language model was smoothed with Knesler-Ney backoff (Kneser and Ney, 1995). The remaining 10% of the corpus was kept in reserve. From the 10%, 1% (66,464 sentences) were taken for testing. From this any sentences with length over 18 words were discarded – the time taken to evaluated longer sentences is too long to be feasible as it increases exponentially.

[1]Available online at `http://nlp.stanford.edu/projects/glove/`

Process	Portion Perfect	BLEU Score	Portion Feasible
Ref. BOW+Ord.	66.6%	0.806	99.6%
Sel. BOW+Ord.	62.2%	0.745	93.7%

Table 1: The overall performance of the Sel. BOW+Ord. sentence generation process when evaluated on the Books corpus.

Process	Portion Perfect	Mean Precision	Mean Jaccard Score
Sel. BOW (only)	75.6%	0.912	0.891

Table 2: The performance of the word selection step, on the Books corpus. This table shows a subset of the results reported by White et al. (2016).

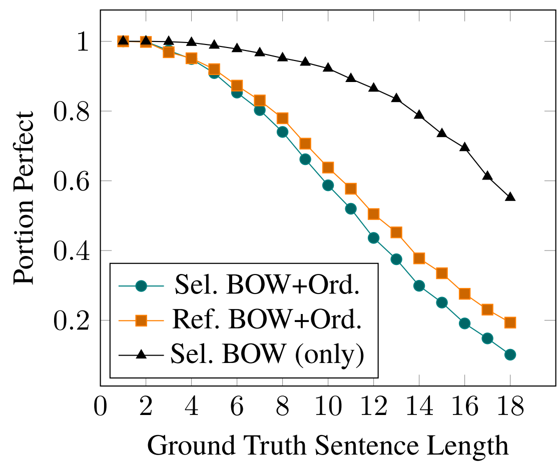

Figure 4: The portion of sentences reconstructed perfectly by the Sel. BOW+Ord. process. Shown also is the results on ordering only (Ref. BOW+Ord.), which orders the reference BOWS; and the results from the word selection step only (Sel. BOW (only)) i.e. the input to the ordering step.

This left a final test set of 53,055 sentences. Figure 3 shows the distribution of the evaluation corpus in terms of sentence length.

Note that the Books corpus contains many duplicate common sentences, as well as many duplicate books: according to the distribution site[2] only 7,087 out of 11,038 original books in the corpus are unique. We did not remove any further duplicates, which means there is a strong chance of a small overlap between the test set, and the set used to fit the trigrams.

4.3 Mixed Integer Programming

Gurobi version 6.5.0 was used to solve the MIP problems, invoked though the JuMP library (Lubin and Dunning, 2015). During preliminary testing we found Gurobi to be significantly faster than the open source GLTK. Particularly for longer sentences, we found two orders of magnitude difference in speed for sentences of length 18. This is inline with the more extensive evaluations of Meindl and Templ (2012). Gurobi was run under default settings, other than being restricted to a single thread. Restricting the solver to a single thread allowed for parallel processing.

Processing was carried out on an AMD Opteron 6300 virtual machine with 45GB of RAM. Implementation was done in the Julia programming language (Bezanson et al., 2014). The implementation, and non-summarised results are available for download.[3]

5 Results and Discussion

The overall results for the two step method (Sel. BOW+Ord.) sentence generation are shown in Table 1. Also shown are the results for just the ordering step, when the reference bag of words in provided as the input (Ref. BOW+Ord.). Table 2 shows the results for the Word Selection step only (Sel. BOW (only)). Both these results place an upper bound on the performance of the overall. The ordering only results (Ref. BOW+Ord.) show the best performance that can be obtained in ordering with this language model, when no mistakes are made in selection. Similarly, the selection only results (Sel. BOW (only)) are bounding as no matter how good the word ordering method is, it cannot recreate perfectly accurate sentences using incorrect words.

It can be noted that Ref. BOW+Ord. and Sel. BOW+Ord. were significantly more accurate than the best results reported by Horvat and Byrne (2014). We attribute this to Horvat and Byrne preprocessing the evaluation corpora to remove the easier sentences with 4 or less words. We did not remove short sentences from the corpus. The performance on these sentences was particularly high, thus bring up the overall results on ordering.

[2] `http://www.cs.toronto.edu/~mbweb/`
[3] [[URL Blinded for Review]]

3A Reference	name this 1922 novel about leopold bloom written by james joyce .	**Sel.**	**Ord.**
Ref. BOW+Ord.	written by name this . novel about 1922 bloom leopold james joyce	–	✖
Sel. BOW+Ord.	written novel by name james about leopold this bloom 1922 joyce .	✓	✖
DT-RAE Ref.	name this 1906 novel about gottlieb_fecknoe inspired by james_joyce		
DT-RAE Para.	what is this william golding novel by its written writer		
3B Reference	ralph waldo emerson dismissed this poet as the jingle man and james russell lowell called him three-fifths genius and two-fifths sheer fudge .	**Sel.**	**Ord.**
Ref. BOW+Ord.	sheer this as james two-fifths emerson fudge lowell poet genius waldo called russell the and ralph and him . dismissed jingle three-fifths man	–	✖
Sel. BOW+Ord.	him “ james great as emerson genius ralph the lowell and sheer waldo three-fifths man fudge dismissed jingle russell two-fifths and gwalchmai 2009 vice-versa ______________________________ prominent called 21.25 explained	✗	✖
DT-RAE Ref.	henry_david_thoreau rejected this author like the tsar boat and imbalance created known good writing and his own death		
DT-RAE Para.	henry_david_thoreau rejected him through their stories to go money well inspired stories to write as her writing		
3C Reference	this is the basis of a comedy of manners first performed in 1892 .	**Sel.**	**Ord.**
Ref. BOW+Ord.	this is the basis of a comedy of manners first performed in 1892 .	–	✓
Sel. BOW+Ord.	this is the basis of a comedy of manners first performed in 1892 .	✓	✓
DT-RAE Ref.	another is the subject of this trilogy of romance most performed in 1874		
DT-RAE Para.	subject of drama from him about romance		
3D Reference	in a third novel a sailor abandons the patna and meets marlow who in another novel meets kurtz in the congo .	**Sel.**	**Ord.**
Ref. BOW+Ord.	kurtz and another meets sailor meets the marlow who abandons a third novel in a novel in the congo in patna .	–	✗
Sel. BOW+Ord.	kurtz and another meets sailor meets the marlow who abandons a third novel in a novel in the congo in patna .	✓	✗
DT-RAE Ref.	during the short book the lady seduces the family and meets cousin he in a novel dies sister from the mr.		
DT-RAE Para.	during book of its author young lady seduces the family to marry old suicide while i marries himself in marriage		
3E Reference	thus she leaves her husband and child for aleksei vronsky but all ends sadly when she leaps in front of a train .	**Sel.**	**Ord.**
Ref. BOW+Ord.	train front of child vronsky but and for leaps thus sadly all her she she in when aleksei husband ends a . leaves	–	✖
Sel. BOW+Ord.	she her all when child for leaves front but and train ends husband aleksei leaps of vronsky in a sadly micro-history thus , she the	✗	✖
DT-RAE Ref.	however she leaves her sister and daughter from former fianc and she ends unfortunately when narrator drives into life of a house		
DT-RAE Para.	leaves the sister of man in this novel		

Table 3: A comparison of the output of the Two Step process proposed in this paper, to the example sentences generated by the DT-RAE method of Iyyer et al. (2014). Ref. BOW+Ord. shows the word ordering step on the reference BOW. the Sel. and Ord. columns indicate if the output had the correct words selected, and ordered respectively. ✖ indicates not only that ordering was not correct, but that the MIP problem had no feasible solutions at all. In DT-RAE Ref. is shown the result of the method of Iyyer et al. (2014), when the dependency tree of the output is provided to the generating process. Where-as in DT-RAE Para. an arbitrary dependency tree of the is provided to the generating process. Note that the reference used as input to Sel. BOW+Ord. and Ref. BOW+Ord. sentence was varied slightly from that used in Iyyer et al. (2014) and White et al. (2016), in that terminating punctuation was not removed, and multiword entity references were not grouped into single tokens.

4A	**Reference**	we looked out at the setting sun .	**Sel.**	**Ord.**
Ref. BOW+Ord.		we looked out at the setting sun .	–	✓
Sel. BOW+Ord.		we looked out at the setting sun .	✓	✓
VAE Mean		they were laughing at the same time .		
VAE Sample1		ill see you in the early morning .		
VAE Sample2		i looked up at the blue sky .		
VAE Sample3		it was down on the dance floor .		
4B	**Reference**	i went to the kitchen .	**Sel.**	**Ord.**
Ref. BOW+Ord.		i went to the kitchen .	–	✓
Sel. BOW+Ord.		i went to the kitchen .	✓	✓
VAE Mean		i went to the kitchen .		
VAE Sample1		i went to my apartment .		
VAE Sample2		i looked around the room .		
VAE Sample3		i turned back to the table .		
4C	**Reference**	how are you doing ?	**Sel.**	**Ord.**
Ref. BOW+Ord.		how are you doing ?	–	✓
Sel. BOW+Ord.		how 're do well ?	✗	✗
VAE Mean		what are you doing ?		
VAE Sample1		are you sure ?		
VAE Sample2		what are you doing, ?		
VAE Sample3		what are you doing ?		

Table 4: A comparison of the output of the Two Step process proposed in this paper, to the example sentences generated by the VAE method of Bowman et al. (2015).

5A	**Reference**	it was the worst of times , it was the best of times .	**Sel.**	**Ord.**
Ref. BOW+Ord.		it was the worst of times , it was the best of times .	–	✓
Sel. BOW+Ord.		it was the best of times , it was the worst of times .	✓	✗
5B	**Reference**	please give me directions from paris to london .	**Sel.**	**Ord.**
Ref. BOW+Ord.		please give me directions to london from paris .	–	✗
Sel. BOW+Ord.		please give me directions to london from paris .	✓	✗

Table 5: A pair of example sentences, where the correct order is particularly ambiguous.

The resynthesis of the two step process (Sel. BOW+Ord.) degrades as the sentence length increases as shown in Figure 4. It can be seen from the figure that this is largely caused by errors in the ordering step, rather than in the selection step. Though the selection failures are responsible for the drop in performance of Sel. BOW+Ord. below that of the Ref. BOW+Ord., this decrease is much less than the number of errors in that occur in Sel. BOW (only) at that word length. This indicates that many of the sentences which failed the word selection step, also would have failed the ordering step even if the words were perfectly selected.

The method is shown to be able to often exactly reproduce sentences based on their embeddings. Due to its exact and near exact resynthesis of whole sentences it is possible to assess it on a whole corpus, rather than having to demonstrate it only on a few examples. For quantitative comparison, the algorithm was also executed on the examples found in existing works.

The sentences shown in Table 3, are difficult. It features long complex sentences, which are high in proper nouns. These examples are sourced from Iyyer et al. (2014), and the output from their DT-RAE method is also show. Only 3C is completed perfectly with our two step process. Of the remainder the MIP word ordering problem has no solutions, except for in 3D, where it is wrong, but does produce an ordered sentence. In all the others the language model has indicated that there is no way to order them. This failure may be attributed in a large part to the proper nouns. Proper nouns are very sparse in any training corpus for language modelling. The Kneser-Ney smoothed trigram only back-off down to bigrams, so if the words of the bigrams from the training corpus never appear adjunctly in the training corpus, ordering fails. This is largely the case for very rare words. The other significant factor is the sentence length.

The sentences in Table 4, are short and use common words – they are easy to resynthesis. These examples come from Bowman et al. (2015), and the output of their VAE based approach can be compared to that of our two step approach. Of the three there were two exact match's and one near match. The near match is interesting, as they are not commonly produced by the Sel. BOW+Ord. process. Normally mistakes made in the word selection step result in an unorderable sentence. Failures in selection are likely to result in BOW that cannot be grammatically combined e.g. missing conjunctions. This results in no feasible solutions to the word ordering problem.

The examples shown in Table 5 highlight sentences where the order is ambiguous – where there are multiple reasonable solutions to the word ordering problem. In both cases the word selection performs perfectly, but the ordering is varied. In 5A, the Ref. BOW+Ord. sentence and the overall Sel. BOW+Ord. sentence differ in word order but not in word content. This is because under the trigram language model both sentences have exactly identical probabilities, so it comes to which solution is found first. This is the only situation where the method is non-deterministic. In 5B the word order is switched – "from paris to london" vs "to london from paris", which has the same meaning. But, it could also have switched the place names. In cases like this where two orderings are reasonable, the ordering method is certain to fail consistently for one of the orderings. Though it is possible to output the second (and third etc.) most probable ordering, which does ameliorate the failure somewhat.

The two step method breaks the selecting the words and ordering them into separate steps. This means that unorderable words can be selected. This is not a problem for the existing methods of Iyyer et al. (2014) and of Bowman et al. (2015). Iyyer et al. (2014) guarantees grammatical correctness, as the syntax tree must be provided at an input for resynthesis – thus key ordering information is indirectly provided and it is generated into. Bowman et al. (2015) on the other hand integrates the language model with the sentence embedding so that every point in the vector space includes information about word order. In general, it seems clear that incorporating knowledge about order, or at least co-occurrence probabilities, should be certain to improve the selection step. Even so the current simple two step approach have strong capacity to get exact reproductions, without such enhancement.

6 Conclusion

A two step method was presented for regenerating sentences, from the sum of a sentence's word embeddings.

The first part of the two step method is the word selection problem, of going from the sum of embeddings to a bag of words. To solve this we utilised the method presented in White et al. (2016). White et. al. presented a greedy algorithm that was found to perform well at regenerating BOW. We extended that method with a second step, to order the words.

The word ordering is carried out by defining a task of finding the most likely sequence of trigrams. This is expressed as a MIP problem. This method is an extension to the graph-based work of Horvat and Byrne (2014). It was demonstrated that a probabilistic language model can be used to order the bag of words output to regenerate the original sentences. While it is certainly impossible to do this perfectly in every case, for many sentences the most likely ordering is correct.

Resynthesis degraded as sentence length increased. White et al. (2016) showed that the word selection degrades with sentence length, and improves with higher dimensional embeddings. Due to their findings, we only evaluated embeddings of 300 dimensions. It was found as expected that the accuracy of ordering also decreases, but remained strong with higher dimensional models up to reasonable length. The technique was only evaluated on sentences with up to 18 words (inclusive), due to computational time limitations. On these it performed quite well achieving perfect recreation in 62.2% of cases. To the author's knowledge our proposed method the first method to report exact recreation of a substantive corpus. Performances, both accuracy and running time worsens as sentence length increases. With that said, short sentences are sufficient for many practical uses.

From a theoretical basis the resolvability of the selection problem shows that adding up the word embeddings does preserve the information on which words were used; particularly for higher dimensional embeddings. This shows clearly that collisions do not occur (at least with frequency) such that two unrelated sentences do not end up with the same SOWE representation.

References

[Bezanson et al.2014] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B. Shah. 2014. Julia: A fresh approach to numerical computing.

[Bowman et al.2015] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. 2015. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*.

[Dinu and Baroni2014] Georgiana Dinu and Marco Baroni. 2014. How to make words with vectors: Phrase generation in distributional semantics. In *Proceedings of ACL*, pages 624–633.

[Farhadi et al.2010] Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. 2010. Every picture tells a story: Generating sentences from images. In *Computer Vision–ECCV 2010*, pages 15–29. Springer.

[Guevara2010] Emiliano Guevara. 2010. A regression model of adjective-noun compositionality in distributional semantics. In *Proceedings of the 2010 Workshop on Geometrical Models of Natural Language Semantics*, pages 33–37. Association for Computational Linguistics.

[Hochreiter and Schmidhuber1997] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

[Horvat and Byrne2014] Matic Horvat and William Byrne. 2014. A graph-based approach to string regeneration. In *EACL*, pages 85–95.

[Iyyer et al.2014] Mohit Iyyer, Jordan Boyd-Graber, and Hal Daumé III. 2014. Generating sentences from semantic vector space representations. In *NIPS Workshop on Learning Semantics*.

[Kågebäck et al.2014] Mikael Kågebäck, Olof Mogren, Nina Tahmasebi, and Devdatt Dubhashi. 2014. Extractive summarization using continuous vector space models. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)@ EACL*, pages 31–39.

[Kingma and Welling2013] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

[Kneser and Ney1995] Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 1, pages 181–184. IEEE.

[Le and Mikolov2014] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196.

[Lubin and Dunning2015] Miles Lubin and Iain Dunning. 2015. Computing in operations research using julia. *INFORMS Journal on Computing*, 27(2):238–248.

[Meindl and Templ2012] Bernhard Meindl and Matthias Templ. 2012. Analysis of commercial and free and open source solvers for linear optimization problems. *Eurostat and Statistics Netherlands*.

[Mitchell and Lapata2008] Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *ACL*, pages 236–244.

[Pennington et al.2014] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1532–1543.

[Ritter et al.2015] Samuel Ritter, Cotie Long, Denis Paperno, Marco Baroni, Matthew Botvinick, and Adele Goldberg. 2015. Leveraging preposition ambiguity to assess compositional distributional models of semantics. *The Fourth Joint Conference on Lexical and Computational Semantics*.

[Socher et al.2011] Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems 24*.

[Socher et al.2014] Richard Socher, Andrej Karpathy, Quoc V Le, Christopher D Manning, and Andrew Y Ng. 2014. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics*, 2:207–218.

[Socher2014] Richard Socher. 2014. *Recursive Deep Learning for Natural Language Processing and Computer Vision*. Ph.D. thesis, Stanford University.

[White et al.2015] Lyndon White, Roberto Togneri, Wei Liu, and Mohammed Bennamoun. 2015. How well sentence embeddings capture meaning. In *Proceedings of the 20th Australasian Document Computing Symposium*, ADCS '15, pages 9:1–9:8, New York, NY, USA. ACM.

[White et al.2016] Lyndon White, Roberto Togneri, Wei Liu, and Mohammed Bennamoun. 2016. Generating bags of words from the sums of their word embeddings. In *17th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*.

[Yogatama et al.2015] Dani Yogatama, Fei Liu, and Noah A Smith. 2015. Extractive summarization by maximizing semantic volume. *Conference on Empirical Methods in Natural Language Processing*.

[Yu and Dredze2015] Mo Yu and Mark Dredze. 2015. Learning composition models for phrase embeddings.

Transactions of the Association for Computational Linguistics, 3:227–242.

[Zanzotto et al.2010] Fabio Massimo Zanzotto, Ioannis Korkontzelos, Francesca Fallucchi, and Suresh Manandhar. 2010. Estimating linear models for compositional distributional semantics. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1263–1271. Association for Computational Linguistics.

[Zhang et al.2014] Jiajun Zhang, Shujie Liu, Mu Li, Ming Zhou, and Chengqing Zong. 2014. Bilingually-constrained phrase embeddings for machine translation. ACL.

[Zhu et al.2015] Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *arXiv preprint arXiv:1506.06724*.