Part I

Introduction

Chapter 1

Introduction

It has been a continual surprise, that simple combinations of embeddings performs so well for a variety of tasks in natural language processing. At first glance, such simple methods capturing only unordered word use should have little capacity in the rich and highly structured nature of human language as we linguistically understand it. However at a second glance, similar surface information has been used in information retrieval with great success since the inception of the field (**maron1961automatic**). Linear combinations of embeddings can be considered as a dimensionality reduction of a bag of words, with various weightings. Dimensionality reduction can be characterised as finding the best lower dimensional representation of an input according to some quality criterion. In the case of word embeddings, that quality criterion is generally along the lines the ability to predict the co-occurring words – an salient quality of lexical semantics. As such, linear combinations of embeddings take bag of words which is a strong surface form representation, take reduce it to a dense representation that captures lexical semantics.

Throughout over the last three years that we have been considering this problem, others also have also found, often to their own surprise, the strength of simple linear combinations of embeddings.

arora2016simple's work describes a "**arora2016simple**", which is a linear combination of word embeddings. There proposed model is a more complicated combination that considered here. But never the least, is primarily a weighted sum of embeddings, with small adjustment based on linear dimensionality reduction methods. In particular when using the word embeddings of **wieting2015towards**, they find this to be very competitive with more complex models which take into account word order.

acl2018bleuopposedmeaning found that taking a mean of word embeddings outperformed almost all their more sophisticated machine-translation-based sentence representations when used on classification and paraphrase detection tasks. This is not to say that linear combinations of embeddings are ideal models for all tasks. They clearly can not truly handle all the complexities of language. But rather that the occurrence of the complexities they can not handle is rarer in practice in many tasks than is often expected.

ac2018probingsentencevectors constructed 10 probing tasks to isolate the some of the information captured by sentence representations. They found the strong performance of averaged word embeddings on sentence level tasks to be striking. They attribute it to the sentence level information being redundantly encoded in the word-forms: the surface level information is surprisingly useful for what at first looks like a sophisticated tasks. With the exception of their word-content task, they did find more sophisticated models able to perform better the the averaged word embeddings. However, when correlating the performance of their probing task against real world tasks, they found that the word-content probing task was by far the most positively correlated with the real word tasks. This makes it clear how valuable surface information is in practical tasks.

In the work presented this dissertation, we find that that even in tasks where it would seem that non-surface information incorporating word-order is required, we find in practice other issues cause the more powerful models that are (theoretically) able to handle these situations correctly may them to be never-the-less outperformed. This is particularly the case where the theoretical improvement from incorporating this information is small, relative to the practical complexity of the techniques required to leverage it. Such a case where where word order matters but the error from ignoring it is small, is particular illustrated in **??**.

At a high-level the success of these techniques comes down to most human language being easy to understand and simple. This expectation of language being easily understood is highlight by (**grice1975logic**), which brings the expectation the communication is conducted following the

cooperative principle The overall supermaxim for Grice's cooperative principle is the speakers should "be perspicuous" or more perspicuously, should use speech that is clearly expressed and easily understood. The particular relevant maxims within the principle are: the maxim of quantity, that one should should make contributions no more nor less informative than required; and the maxim of manner: to avoid ambiguity and obscurity of expression, and to make contributions that are brief and orderly. While Grice originally proposed these are exceptions upon conversation, the general principle applies more broadly to natural language communication. This general principle being that language used is normally expected to be understood easily – thus fulfilling the goal of communicating.

Adversarial examples are reasonably easy to construct. An adversarial example to a linear combination of word embeddings is any text where the word order significantly effects that meaning; and where multiple possible word orders exist. For such an adversary to be significant, both word orders must be reasonably likely to occur. However; such cases are rarer than one might expect, as was found indirectly in **??**. Particularly when punctuation is included, which it reasonably can be as a token embedding. As such, while these cases certainly exist, we find that for real applications they are sufficiently rare that the simplicity of the linear combinations of embeddings type approach can work very well.

The when applied in sentence/phrase representation contexts, such as discussed in **??**, and **??** this gives support to the notion that word order is often not a very significant feature in determining meaning. It seems clear that word order, and other factors of linguistic structure must contribute to the meaning of the phrase. However, our result suggest that it is often in a minor way, and that for many tasks these linear combinations are superior due to their simplicity and effectiveness. While taking into account greater linguistic structure may be the key to bridging the between "almost perfect" and "true perfection", the current state of the field for many tasks has not reached "almost perfect", and as such simpler methods still form an important part.

To further understand the relationship between SOWE and BOW, and the extent to which word order matters the capacity to reverse the conversion from phrase to SOWE in investigated in **??** and **??**. The results in **??** show that it is indeed largely possible to reconstruct bags of words from SOWE, suggesting that when considered as a a dimensionality reduction technique SOWE does not lose much information. This is extended in **??** to order those bags of words back to sentences via a simple tri-gram language model. This had some success at outright reconstructing the sentences. This highlights the idea that for many bags of words (which can be reconstructed form a sum of word embeddings) there may truly be only one reasonable sentence from which they might have come. This would explain why SOWE, and BOW, ignorance of word order does not prevent them from being useful representation of sentence.

The successes of the sums of word embeddings discussed in **??**, and **??** leads us to consider other uses of linear combinations for representation. **??** and **??** consider tasks well outside of phrase representation where the order clearly does not matter.

On the complexity of models. One of the attractive features of these linear combinations is there simplicity This is true both in an implementation sense, and in the sense of gradient descent. For example, the vanishing gradient problem in deep networks, especially RNNs and RvNNs simply does not exist for a sum of word embeddings due to it not being as an input structure. This in contrast to RNNs which are deep in time, and RvNNs which are deep in structure. Deep networks can be placed upon the input processing as represented by a RNN, RvNN or linear combination of embeddings, but for the RNN, and RvNN the network is already depth even with only one hidden layer on top.

TODO: INSERT RNN FAIL PAPER REFERENCE HERE

1.1 Thesis Overview

This thesis tackles a number of natural language understanding problems, and in the solutions draws conclusions on the capacity of linear combinations of embeddings.

The representation of *sentences* is investigated in **??**, through a paraphrase grouping tasks. Similarly, the representation of *phrases* is investigated in **??** through a color understanding (estimation) task. Given the observed properties found by sums of word embeddings, this leads to the investigation of if weighted sums of word sense embeddings might better resplendent a particular usage of a word in **??**. The capacity also lends to the investigation of using a sum of word embeddings to represent the contexts of all usages of a named entity, for the point of view character detection task investigated in **??**. We conclude with a complementary pair of works in **????**, which investigate the ability to recover bags of words and sentences, from sums of word embeddings rep-

Chapter	Structure	Task	Embeddings
??	Sentences	Paraphrase grouping	Word2Vec (**mikolovSkip**)
??	Short Phrases	Color understanding	FastText (**bojanowski2016enriching**)
??	Word Senses	Similarity with context & Word sense disambiguation	AdaGram (**AdaGrams**) & Bespoke greedy sense embeddings
??	Adj. Contexts	POV character detection	FastText (**bojanowski2016enriching**)
??	Sentences	Recovering bags of words	GLoVE (**pennington2014glove**)
??	Sentences	Recovering sentences	GLoVE (**pennington2014glove**)

Table 1.1: Summary of the investigations published within this dissertation.

resenting sentences. These final works illustrate some of the reasons why the linear combinations work so well.

1.1.1 ?? White2015SentVecMeaning (White2015SentVecMeaning)

We begin by examining methods for representing sentences. Sentences are a fundamental unit of communication – a sentence is a single complete idea.

The core goal is to determine if an sentence embedding clearly separated the different ideas. Paraphrases are defined by a bidirectional entailment relationship between two sentences. This is a equivalence relationship, it thus gives rise to a partitioning of all sentences in natural language space. If a sentence embedding is of high quality, it will be easy to define a corresponding partitioning of the embedding space. One way to determine how easy it is to define the corresponding partitioning it to attempt to do just that as a supervised classification task using a weak classifier. A weak classifier, such as the linear SVM we used, is required as a more powerful classifier (such as a deep neural network) could learn arbitrary transforms. The classification task is to take in a sentence embedding and predict which group of paraphrases it belongs to. Where the target paraphrase group is defined using other paraphrases with the same meaning as the candidate.

Under this course of evaluation it was found that the sum and mean of word embeddings performed very well as a sentence representation. These LCOWEs were the best performing models under evaluation. They were closely followed by the bag of words, which is advantaged by being much higher dimensionality than other models. The LCOWEs outperform the bag of words as they also capture synonyms and other features of lexical relatedness. Slightly worse than the bag of words was the bag of words with PCA dimensionality reduction to 300 dimensions. This confirms our expectation that LCOWEs are a better form of dimensionality reduction for preserving meaning from a bag of words than PCA.

The poor results of the paragraph vector models (**le2014distributed**) is in line with the observation in the footnotes of the less well-known follow up work **mesnil2014ensemble**. That the performance reported **le2014distributed** can not be reliably repeated on other tasks, or even the same takes with a slightly different implementation.

A limitation of this work is that it does not include the examination of any encoder-decoder based methods, such as Skip-Thought (**DBLP:journals/corr/KirosZSZTUF15**), or machine translation models. Another limitation of the work is that the URAE use a pretrained model with only 200 dimensions, rather than 300 dimensions as was used in the other evaluations.

Paraphrases provide one source of grounding for evaluation of sentences. Color names are a subset of short phrases which also have a ground truth for meaning – the color. They are thus useful for evaluating the performance of LICOWE on short phrases.

1.1.2 ?? White2018ColorEst (White2018ColorEst)

To evaluated the performance of input representations for short phrases, we considered a color understanding task. Color understanding is considered as grounded microcosm of natural language understanding (**2016arXiv160603821M**). It appears as a complicated sub-domain, with many of the same issues that plague natural language understanding in general: it features a lot of ambiguity, substantial morphological and syntax structure, and depends significantly on context that is not made available to the natural language understanding algorithms. Unlike natural

language more generally, it has a comparatively small vocabulary, and it has grounded meaning. The meaning of a particular utterance, say `bluish green`, can be grounded to a point in color space, say in HSV (192°, 93%, 72%), based on the questioning the speaker. The general meaning of the a color phrase can be grounded to a distribution over color space, based on surveying the population of speakers.

Models were thus created to learn a mapping from natural language space, to points or distributions in color space. Three input representations were considered: a sum of word embeddings (SOWE), a convolutional neural network (CNN), and a recurrent neural network (RNN). The SOWE corresponds to a bag of words – no knowledge of order. The CNN corresponds to a bag of ngrams – it includes features of all length, thus can encode order. The RNN is a fully sequential model – all inputs are processed in order and it must remember previous inputs.

It was expected that this task would benefit significantly from a knowledge of word order. For example, `bluish green` and `greenish blue` are visibly different colors. The former being greener than the later. However, it was found that the SOWE was the best performing input representation, followed closely by the CNN , with the RNN performing much worse. This was even the case when the test set was restricted to only contain colors names for which multiple different word orders (representing different colors) were found in the training set. This can be attributed to the difficulty in training the more complicated models. In contrast to a simple feed-forward SOWE, in a RNN the gradient must propagate further from the output, and there is more weights to be learned in the gates. This difficulty dominated over the limitation in being able to model the color names correctly. We note that while `bluish green` and `greenish blue` are different colors, but they are never the less similar colors. As such, the error from treating them as the same, is less than the error caused by training difficulties.

The solving problem of color estimation from natural language color name, has pragmatic uses. Color estimation from description has utility as a tool for improving human-computer interaction. For example allowing free(-er) text for specifying colors in plotting software, using point estimation. It also has utility as an education tool: people from different cultures, especially non-native English speakers, may not know exactly what color range is described by `dark salmon`, and our model allows for tools to be created to answer such queries using distribution estimation.

A limitation of this study is the metrics used. For distribution estimation, the perplexity of the discretized distributions in color space is reported. It would be preferable to uses Kullback–Leibler divergence, which would allow comparisons to future works that output truly continuous distributions. Kullback–Leibler divergence is monotonically related the to discretized perplexity, however. For point estimation, using an evaluation metric such as a Delta-E, which is controlled for the varying sensitivity of human perception for different hues. Neither limitation has direct bearing on the assessment of the input representations.

1.1.3 ?? WhiteRefittingSenses (WhiteRefittingSenses)

With the demonstrated utility of linear combinations of embeddings for representing the meanings of larger structures made from words, it is worth investigating their utility for representing the possible different meanings of words. When it comes to representing word senses, it may be desirable to find a representation for the exact sense of a word being used in a particular example. A a very fine grained word sense for just that one use. If one has a collection of induced word senses, it seems reasonable to believe that the ideal word sense for a particular use, must lay somewhere between them in embedding space. Further more, if one knows the probably of each of the coarse induced senses being the correct sense for this use, then it makes sense that the location of the fine grained sense embedding would be closer to the more likely coarse sense, and further from the less likely coarse sense. As such we propose a method to define these specific case word senses based on a probability weighted sum of coarser word sense embeddings. We say that we *refit* the original sense embeddings, using the single example sentence to induce the fine grained sense embedding.

Using this we define a similarity measure which we call RefittedSim, which we find to work better than AvgSimC (**Reisinger2010**). AvgSimC is a probability-weighted average of all the pairwise similarity scores for each sense embedding. In contrast RefittedSim is a single similarity score as measured between the two refitted vectors – which are the probability weighted averages of the coarser sense vectors. On the embeddings used in our evaluations this gave a solid improvement over AvgSimC. It is also asymptotically faster to evaluate.

We also evaluated using refitting for word sense disambiguation (WSD). Normally, induced senses can not be used for word sense disambiguation, as they do not correspond to standard dictionary word senses. By using the WordNet gloss (definition) as an example sentence, we are able to use refitting to create a new set of sense embeddings suitable for WSD. Using this we

can use the skip-gram formulation for probability of the context given the refitted sense, and so apply Baye's theorem to find the most-likely sense. However, we found that the results were only marginally better than the baseline. Nearly unsupervised WSD is a very difficult problem; with a strong baseline of simply reporting the most-common sense. Our results do suggest that our refitting method does not learn features that are antithetical to WSD. However, they do incorporate the most frequent sense as a prior and seem to provide little benefit beyond that.

A limitation of this study was that it did not perform the evaluation on state-of-the-art word-sense embeddings. As such, while it's comparisons between these embeddings are valid they can not be readily compared to the current state-of-the-art on the tasks.

1.1.4 ?? novelperspective (novelperspective)

Given the success of LCOWEs for representing meaningful linquistic structures (sentences and phrases), a natural follow up question is on its capacity to represent combinations of words that do not feature this natural kind of structure. These would be more arbitrary bags of words; that never the less may be useful features for a particular task. The task investigated in this work was about identifying point of view characters in a novel.

Given some literary text written in third person limited point of view, such as Robert Jordan's popular *"Wheel of Time"* series of novels, it is useful to a reader (or person analysing the text), to identify which sections are from the perspective of which character. That is to say, we would like to classify the chapters of a book according to which character they are from the perspective of. This at first looks like a multiclass classification problem; however it is in-fact an information extraction problem. The set of possible classes for any given chapter is the set of all named entities in the book. Different booked have different characters, thus the set of named entities in the training data will not match that of an arbitrary book selected by a user. As such, the named entity tokens themselves can not be used in training for this task. Instead, it must be determined whether or not a named entity is the point of view character, based on how the named entity token is used. To do this, an representation of the context of use is needed.

The task can be treated as a binary classification problem. Given some feature vector representing how a particular named entity token was used throughout a chapter, find the probability of that named entity being the point of view character. We considered two possible feature sets to use to generate the feature vectors for named entity token use. Both feature sets consider the context primarily in terms of the token (word) immediately prior to, and the token (word) immediately after the named entity. We define a 200 dimensional hand-crafted *classical feature set* in terms of the counts of adjacent part of speech tags, position in the text, and token frequency. We define a *mean of word embedding based feature set* as the concatenation of the mean of the word embedding for the words occurring immediately prior, to the mean of the word occurring immediately after. As this was using 300 dimensional embeddings, this gives a 600 dimensional feature vector.

It was found that the two feature sets performed similarly, with both working very well. It seems like the primary difficulty was with the high dimensionality of the word embedding based feature set. Without sufficient training data, it over-fit quiet easily. It's performance dropped sharply on the testset, compared to it's oracle performance if trained on the testset, when the largest book series was removed. This likely could have been ameliorated by using lower dimensional embeddings.

The good performance of the word embedding based feature set is surprising here as it does not include any frequency information. We used a mean, rather than a sum, of word embeddings to represent the context of named entity token use. In the classical feature set, we found that by far the most important feature was how often that named entity token was used. Indeed just reporting the most frequently mentioned named entity gave a very strong baseline. The lexical information captured by the MOWE is clearly similarly useful to the part of speech tag counts, and almost certainly makes more fine grained information available to the classifier. Thus allowing it to define good decisions boundaries for if the feature vector represents a point of view character or not.

A limitation of this study is that different binary classifiers were used for the two feature sets. Ideally, the performance using a range of classifiers for both would have been reported. Our preliminary results suggested that the classifier choice was not significant. With logistic regression, SVM, and decision trees giving similarly high results for both feature sets.

1.1.5 ?? White2015BOWgen (White2015BOWgen)

Given the consideration of a sum of word embeddings as dimensionality reduced form of a bag of words a important question is to how recoverable the bag of words is from the sum. A practical

way to lower bound the loss of information is to demonstrate a deterministic method that can recover a portion of the bag of words.

We propose as method to extract the original bag of words from a sum of word embeddings. Thus placing a bound on the information lost in the transformation of BOW to SOWE. This is done via a simple greedy algorithm with a correction step. The core of this method functions by iteratively searching the vocabulary of word embeddings for the nearest embedding to the sum, adding it's word to the bag of words and subtracting its embedding from the sum. It is thus only computationally viable with reasonably small vocabularies. This method works as each component word in the sum has a unique directional contribution in the high dimensional space. As one would expect, the works better the higher dimensional the embeddings are, and with fewer words. Even with relatively low dimensions it works quiet well. This shows use that embeddings are not for example constantly cancelling each other in the sum.

An interesting alternative to this deterministic method would be to train a supervised model to project from SOWE to a fuzzy bag of words. This is similar to the word-content task considered by **adi2017Probing**. In that task a binary classifier was trained to take a sentence representation and a word embedding for a single word that may or may not appear in the sentence.

1.1.6 ?? White2016a (White2016a)

Given that it was demonstrated that the bag of word can be recovered, the obvious follow up question is if we can recover the the sentence. This **??** is a small supplement to **??**.

The word ordering problem tackled is given a bag of words, and a trigram language model determine the most-likely order for words. This allows bags of words to be turned into the most likely sentences. We define a deterministic algorithm to solve this using linear mixed integer programming. Using this algorithm we can use the partially recovered bags of words from **??** and determine how frequently they can be correctly ordered to find the original sentence.

We find that surprisingly often they can. The majority of sentences of length up to 18 can be successfully recovered from a SOWE. With, as expected, the longer the sentence the more difficult the recovery. This suggests that the number of likely possible orderings for the words in a sentence is much lower than it may at first seem. Particularly since this method based on a simple trigram language does so well. It no doubt a more sophisticated neural network based language model would be significantly better.

The algorithm used in our method is a minor extension of that of **Horvat2014**. We take advantage of the slight differences between the word ordering problem and the generalised asymmetric travelling sales man problem. We can eliminate some branches that would not be possibly for a travelling salesman solver; by directly defining it as a mixed integer linear programming problem.

The use of the methods of **??** and **??**, together with a system trained to output a approximation to a SOWE, is an interesting, though not really practical, method for natural language generation.

1.2 Some Math

1.2.1 Word Embeddings

Given a word represented by an integer w, from a vocabulary $\mathbb{V} \subset \mathbb{Z}$, and a matrix of embeddings, represented as C: its embedding can be found by slicing out the wth column: $C_{:,w}$. For $\tilde{e}_w$ the elementary unit vector, i.e. the one-hot vector representation of w it can be seen that the word embedding can be represented as the product of the embedding matrix with the one-hot vector.

$$C_{:,w} = C\,\tilde{e}_w$$

1.2.2 Sum of Word Embeddings

For some sequence of (not necessarily unique words) words $\mathcal{W} \in \mathbb{V}^{\mathbb{Z}_0}$ represented $\mathcal{W} = \left(w^1, w^2, \ldots, w^n\right)$, where w^i is an integer representing which word the ith word is.

The sum of word embeddings (SOWE) representation is written as:

$$\sum_{i=1}^{i=n} C_{:,w^i}$$

The bag of word (BOW) representation is written as a vector from $\mathbb{Z}^{|\mathbb{V}|}$.

$$\tilde{x} = \sum_{i=1}^{i=n} \tilde{e}_{w^i}$$

,

Using this the sum of word embeddings can be seen to be the product of the embedding matrix with a BOW vector.

$$\sum_{i=1}^{i=n} C_{:,w^i} = \sum_{i=1}^{i=n} C\,\tilde{e}_{w^i} = C \sum_{i=1}^{i=n} \tilde{e}_{w^i} = C\,\tilde{x}$$

1.2.3 Mean of Word Embeddings

The mean of word embeddings representation is written as:

$$\frac{1}{n} \sum_{i=1}^{i=n} C_{:,w^i}$$

Note that n is equal to the element-wise sum of the BOW vector (x), i.e. to its l1-norm:

$$n = \|\tilde{x}\|_1 = \sum_{\forall j \in \mathbb{V}} \tilde{x}_j$$

Thus the mean of word embeddings can be seen as the product of the embedding matrix with the l1-normalized BOW vector.

$$\frac{1}{n} \sum_{i=1}^{i=n} C_{:,w^i} = \frac{1}{n} \sum_{i=1}^{i=n} C\,\tilde{e}_{w^i} = \frac{1}{n} C \sum_{i=1}^{i=n} x = C\,\frac{\tilde{x}}{\|\tilde{x}\|_1}$$

1.2.4 Linear Combination of Embeddings

The full generalisation of this is that any linear combination of embeddings can be seen as product of the embedding matrix, the a weighted bag of words.

A weighting function for linear combination scheme can be defined, mapping from a given bag of words, and a word, to the weighting of that word. $\alpha : \mathbb{Z}^{|\mathbb{V}|} \times \mathbb{V} \to \mathbb{R}$.

(For example for the mean of word embeddings $\alpha(\tilde{x}, w) = \frac{1}{\|\tilde{x}\|_1}$.)

From the weighting function, we can evaluated it for a given BOW, for each word in the vocabulary to define a weighting vector $\tilde{\alpha}^{\tilde{\mathrm{x}}}$:

$$\tilde{\alpha}^{\tilde{\mathrm{x}}} = [\alpha(\tilde{x}, w)]_{w \in \mathbb{V}}$$

Using $\odot$ as the Hadamard (i.e. element-wise) product, we can thus write:

$$\sum_{i=1}^{i=n} \alpha(\tilde{x}, w^i) C_{:,w^i} = C \sum_{i=1}^{i=n} \alpha(\tilde{x}, w^i) \tilde{e}_{w^i} = C\,\left(\tilde{\alpha}^{\tilde{\mathrm{x}}} \odot \tilde{x}\right)$$