A Survey on Distributed Representations of Words

Tanvi Sahay
tsahay@cs.umass.edu

Sri Sudhamsu Krishna Manne
smanne@cs.umass.edu

Abstract

A real valued representation of lexical entities, based on semantic, grammatical or syntactic information, is a popular way to encode information contained in them. These encoded 'embeddings' can then be used as standalone features for tasks such as recognizing similar words, creating word clusters and analyzing parent-child relationships or act as additional features for other NLP tasks. In this survey, we take a look at the different methods in which words can be represented as vectors of real valued elements and how these methods have evolved over time. We focus primarily on distributed representations for words and compare and contrast the different popular approaches for the same. We also discuss certain properties of distributed representations of words which make them useful for a wide variety of NLP tasks.

1 Introduction

The idea of creating word vectors that can capture information pertaining to similarity of words stems from psychology, where in 1952, Charles Osgood first represented words in term of vectors of numbers that corresponded to human attitudinal ratings along a seven-point scale. Today, obtaining meaningful representations of lexical entities such as words, phrases and sentences has become a major area of research in Natural Language Processing. This is primarily because word representations that capture semantic and syntactic information have been shown to improve the performance of models on downstream NLP tasks such as word-sense disambiguation, language modeling, POS tagging and named entity recognition, in addition to showing massive improvements in stand-alone tasks such as finding similar words and concepts in the underlying data.

There are several design choices that can be made when choosing an appropriate representation for words in computational natural language processing tasks. One of the simplest ways is to simply assign an index to each word in the vocabulary. Such representations can also take the form of one-hot vectors – which are of a size equal to the size of the vocabulary and contain zero-valued elements everywhere except a one-valued element at the appropriate index of the word. Such representations have been used to develop several NLP algorithms. But, owing to the large vocabularies required to learn meaningful models, this method of representing words quickly becomes a handicap. The representations begin to get sparser, while taking increasing amount of memory. In addition to being undesirably sparse, they also do not contain any "meaning" or represent the word in any way other than as a unique identifier.

Another way to represent words is to assign them low-dimensional integer or real-valued vectors (low-dimensional w.r.t. the vocabulary size). Such vectors can be dense and avoid all the sparsity issues of one-hot vectors. The dimensions of these vector may be hand-crafted to represent meaning, or they might be learned from data. This survey is concerned with such distributed representations of words.

In recent years, a lot of research work in natural language processing is being dedicated to obtaining meaningful distributed representations of words, phrases, and sentences. These representa-

(March 21, 2018) This Literature Review was submitted as part of COMPSCI 690N Advanced Natural Language Processing (Spring 2018) course at University of Massachusetts Amherst.

tions, or “embeddings”, have been found to encapsulate word meanings – similar or related words can be found close to each other in the multi-dimensional embedding space. Naturally, they have been used in a variety of NLP tasks with great success and have led to wild improvements over the older word representations.

In this survey, we’ll discuss how these distributed representations have evolved over past few decades, from non-connectionist approaches to connectionist approaches. Later, we’ll discuss some properties of such representations which make them particularly useful for NLP tasks. But first, it would be beneficial to revisit the local representations and their disadvantages.

2 Local Representations

Given an element and a set of values to represent it with, the simplest way to represent the elements in a distinct way is to assign a single value to each element. This can be thought of as a “local representation” of the element, since this representation does not take information from any other element into account. One of the most popular techniques of local representation of words, introduced in 1985 and then republished in 1988, was given by Waltz and Pollack (1988). In this paper, they introduced the idea of "micro-features" to perform natural language interpretation using connectionist models. These microfeatures were hand made features that represented quantities that humans used to make distinctions in the real world. This included features such as ’second’, ’minute’, ’hour’, ’factory’, ’office’, ’school’ etc. Each concept was represented as a vector, where each position in the vector corresponded to a microfeature and the numerical value at that position quantified the association of the feature with that concept. These features were defined independently for each concept and hence had no interaction with other concepts, thus making them "local" in nature. While this idea of microfeatures was a better representation for words that a one-hot representation, it was both labor intensive and subjective to human knowledge. While these feature representations were highly interpretable, creating them was time consuming and lack of connection between the different concepts also resulted in loss of information about the individual concept.

Hinton et al. (1986) discussed the drawbacks of localist representations and endorsed advantages to using distributed representations, which changed the course of word representations towards distributed ones. He showed that distributed representations were better at finding similarities and generalizing relationships, since learning didn’t start from scratch for every new element, as it did in local representations. Today, the only local representations in use are one-hot encodings, and are used as inputs to different neural network models that make use of these encodings to extract a distributed representation for the corresponding words. In the next section, we will discuss more about the distributed representations and how these methods changed with improvements in technology.

3 Distributed Representations

Hinton et al. (1986) provided the following definition of distributed representations, that is still relevant today: distributed representations are representations of entities in which each entity is represented by several computing elements and each element is involved in the representation of several entities. These representations are today understood as an integral part of all connectionist networks, otherwise known an neural networks. However, using neural networks for word representations was not always a popular mechanism. In general, distributed representation techniques can be divided into two categories - connectionist approaches and non-connectionist approaches. Each of these categories is explored in more detail in the following sections.

3.1 Non-Connectionist Models of Distributed Representations

Rumelhart et al. (1986) introduced back-propagation as a potential method of extracting distributed representations for entities. However, lack of evidence on how these representations can capture complex relationships in the underlying data was still unseen, and researchers sought other ways of preparing distributed representations for words, without the use of connectionist models.

3.1.1 Context Vectors

Developing the idea of "micro-features" further, Gallant (1991) introduced context vectors, that were defined as vector representations of words derived from the context of that word and used these to disambiguate word senses. A major contribution of this paper was the context algorithm, which was a technique introduced do allow dynamic computation of context vectors for any position in the text. The paper defined a feature space consisting of *n*, manually chosen concepts that were used to outline contexts. Each word was defined as a vector, where each position in the vector belonged to one of the concept words and the numerical value at the position corresponded to the association of the word with that concept. Context vectors for 100000 words were created by hand, similar to Waltz and Pollack (1988). Figure 1 shows a sample of context vectors prepared for the different meanings of the same word, "star". Then, Bootstrapping was used to create dynamic context vectors, by taking into account both the dynamic context vector at the end of the previous sentence and the words present in the current sentence. This algorithm allowed automatic creation of context vectors, thus eliminating the need for hand-made representations.

$$V^{\text{star-sky}} = \left< \begin{array}{rrrrr} -1 & -1 & -1 & -1 & 0 \\ 0 & +2 & +1 & +2 & 0 \\ 0 & 0 & +1 & -1 & -2 \\ +2 & 0 & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{array} \right>$$

$$V^{\text{star-Hollywood}} = \left< \begin{array}{rrrrr} +2 & +1 & +1 & -1 & +1 \\ +2 & -1 & +1 & +2 & +2 \\ +1 & +1 & +1 & +2 & +2 \\ 0 & +2 & +1 & +2 & -1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{array} \right>$$

Figure 1: Context Vectors for different meanings of the word ‘star’. (Gallant, 1991)

Gallant et al. (1992) extended this idea to introduce a system called MatchPlus, that, in addition to providing representations for word stems, also made use of these to prepare document and query vectors in the same representation space, to allow efficient information retrieval. However, while the MatchPlus created context vectors derived from local context information, they ignored any information available outside their context window.

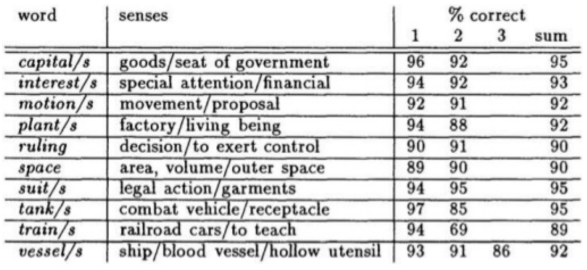
Caid et al. (1995) made use of an already popular technique called Latent Semantic Indexing (Deerwester et al., 1990), in comparison with MatchPlus, for the purpose of document retrieval. Latent Semantic Indexing or LSI prepared a term-document frequency matrix, where each cell in the matrix contained information about the number of times a word had occurred in a document. A matrix factorization technique like SVD was applied on this term-document matrix and only the top k largest independent linear components of the matrix were kept. This resulted in k dimensional document vectors, along with k dimensional word vectors, that were then used to produce query vectors, for document retrieval.

Each of the methods described above presented a unique "context vector" for each word in the document. However, while LSI took only global or document-level co-occurrence into account, MatchPlus gave a different representation to different instances of the same word, based on their different contexts, which made it difficult to compare different concepts, as could be done with "microfeatures". This created the need for a method that could provide unique representations to words, such that different concepts could be compared with each other.

3.1.2 Word Spaces

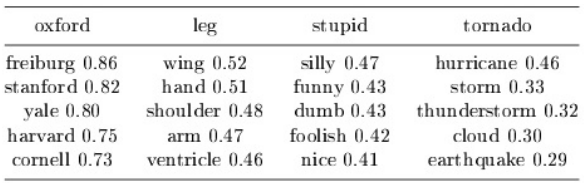
Schütze (1993) introduced a method of extracting word representations that exploited the lexical co-occurrence statistics and allowed a unique representations for words, by taking all instances of a word into account. This model, known as the word space model, prepared a collocation matrix from 5000 popular four-grams and for each new word, prepared a context vector based on the co-occurrence of the word with each four-gram, summed over all instances of the word. This method had the advantage of being completely automated and experimental results showed its ability to perform well on the word sense disambiguation task. Figure 2 shows results for 10 experimental disambiguation tasks, where the columns indicate success for each sense of the particular words. It also gave a unique vector for each word in the document vocabulary, which was an advantage over the context vector approaches. However, the word representations provided by this model were not interpretable on their own and the only information encoded by the model was vector similarity - similar words were closer together in the word space.

Despite the lack of interpretability of word space models, the idea that similar words should be close to each other in a vector space soon became popular. Qiu and Frei (1993) carried the idea of word spaces forward and constructed a similarity thesaurus between words, by taking into account the information of how often words occur in particular documents. Each word representation in this model was a vector of document weights,



word	senses	% correct			
		1	2	3	sum
capital/s	goods/seat of government	96	92		95
interest/s	special attention/financial	94	92		93
motion/s	movement/proposal	92	91		92
plant/s	factory/living being	94	88		92
ruling	decision/to exert control	90	91		90
space	area, volume/outer space	89	90		90
suit/s	legal action/garments	94	95		95
tank/s	combat vehicle/receptacle	97	85		95
train/s	railroad cars/to teach	94	69		89
vessel/s	ship/blood vessel/hollow utensil	93	91	86	92

Figure 2: Results for word sense disambiguation using word space model. (Schütze, 1993)



oxford	leg	stupid	tornado
freiburg 0.86	wing 0.52	silly 0.47	hurricane 0.46
stanford 0.82	hand 0.51	funny 0.43	storm 0.33
yale 0.80	shoulder 0.48	dumb 0.43	thunderstorm 0.32
harvard 0.75	arm 0.47	foolish 0.42	cloud 0.30
cornell 0.73	ventricle 0.46	nice 0.41	earthquake 0.29

Figure 3: Examples of the five Nearest Neighbors to target words using Random Indexing. The numbers indicate the degree of correlation to the target word. (Lund and Burgess, 1996)

where the weight of each document were computed with the help of the feature frequency, the inverse item frequency and maximum feature frequency, features being the documents themselves. This normalized tf-idf weighted matrix presented the advantage that if words co-occurred in longer documents, their assigned similarity would be lower than if they co-occurred in a shorter document. This model took more that just term-term co-occurrence information into account, as was done in LSI, and thus provided smarter word representations. However, the model was now looking at each word on a document level.

The idea that local co-occurrence is a stronger indicator of similarity was exploited by Lund and Burgess (1996) in their HAL (hyperspace analogue to language) method. This method computed co-occurrence statistics, with the strength of association decaying with distance. A window of fixed size was passed over the corpus and for each center word, words lying inside the window were considered as co-occurring words, with the strength of co-occurrence decaying with increasing distance. This was extended to the entire corpus and a co-occurrence matrix was produced. Each axis of the matrix was equal to vocabulary size and the final word vector was a concatenation of both row and column corresponding to a particular word. With these word vectors, the authors performed several experiments to establish their nature. They showed that word vectors exhibited both semantic and associative relationship by analyzing closest neighbors, that the word vectors contained information that allowed categorization of similar groups, such as animals and humans etc. and that the semantic distance between words correlated with human reaction times in a lexical priming study.

A similar representation was later also used by Blitzer et al. (2005), where word representations were extracted by creating a bigram frequency matrix and computing the normalized frequency for all words. These normalized vectors acted as preliminary vectors, on which dimensionality reduction techniques such as PCA and SDE were applied and the final reduced vectors were used as word vectors for a statistical language modeling task.

3.1.3 Random Indexing

Another relatively disparate approach to forming distributed word representations was taken by Sahlgren (2001). In this paper, the author introduced a method called Random Indexing (RI), which was inspired by the HAL model (Lund and Burgess, 1996). In this method, each word was assigned a sparse random vector called a random label, which had a small number of randomly distributed +1s and -1s, with the remaining elements 0. The context vector for each element was computed by adding the labels of all words in the target's context window to its own context vector, for all occurrences of the word. The addition was weighted, with context words father away from target getting a smaller weight. This model was analyzed on a synonym finding task, whose results were compared with a set of TOEFL solutions, on which it performance was comparable to LSI. Figure 3 shows a sample of the nearest neighbors with associated correlation, which are fairly correlated to their corresponding targets.

All the methods explained above used non-connectionist models to extract word representations from the data. In other words, none of the models explained above made use of a neural network model to get word vectors. This was mainly due to lack of evidence regarding the extent to which information can be learned by neural representations. The non-connectionist models presented several advantages and showed good per-

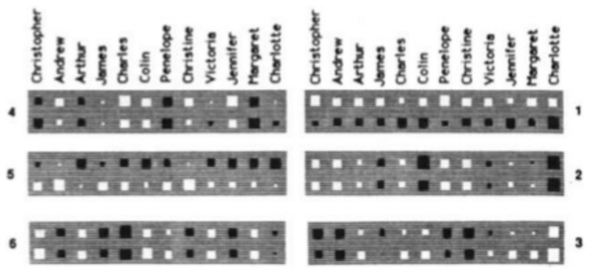


Figure 4: Neuron Activation from 24 input units given to 6 hidden units, corresponding to English and Italian names. (Rumelhart et al., 1988)

formance on multiple NLP tasks. However, the models could not be scaled to incorporate the increasing amount of data, since most of them required the storage of term-term or term-document frequency statistics. While the context vector models still required some amount of manual labor, the word space models had large space complexities and only took lexical statistics into account, thus causing loss of information such as sequence of words, subject verb information, part of speech tags etc. Due to these limitation, neural network based approaches gained popularity and have today become the most commonly used models for extracting distributed representations.

3.2 Connectionist or Neural Network-based Models of Distributed Representations

The first hint that distributed representations learned by neural networks can be used as vector representations for input entities, was provided by Rumelhart et al. (1988), where the authors introduced back-propagation as a method for learning weights in a simple two layer feed-forward neural network. In this paper, the authors showed how neuron activations corresponding to each input can be used as their representation. Figure 4 shows the original image presented in that paper, that shows how activations for every hidden neuron can be collected into a vector, and be treated as distributed representations for an input word. In the figure, white rectangles show excitatory weights and black rectangles show inhibitory weights and sizes represent the weight values.

While the paper presented a novel idea, their lack of evidence for whether connectionist models really capture complex structures in data and whether distributed representations being used by them are able to learn inherent information in the data kept researchers from exploring this technique in more detail. After the introduction of back-propagation however, many works explored the above question, in an attempt to establish the nature of the distributed representations learned by neural networks. One of the most notable was by Hinton (1990) in which the he tried to show that connectionist models that make use of distributed representations can indeed learn complex structures by training a network to capture learn part-whole hierarchies. However, this work did not discuss the quality of representations themselves.

Elman (1991) analyzed representations learned by a recurrent neural network, trained to predict the next word given a previous word for short 2-3 word sentences. The network presented in the paper used an extra hidden layer, which they referred to as a layer of context units, that stored hidden layer activations from the previous time step, and combined these values with the current time step input, to produce new activations. After training the model, final representation for each word was taken as the average of its hidden unit activation at test time, over all occurrences of the word. The paper utilized several mechanisms to explore how the learned distributed representations encode information about the language and underlying data. However, while they suggested that performing hierarchical clustering on the learned representations resulted in groups of nouns and verbs together, they performed no further experimentation to provide proof for whether these representations could be used as word vectors or not.

After a shift in focus of the wider research community back to non-connectionist models from connectionist models, Bengio et al. (2003) reintroduced neural networks as a force to be reckoned with in natural language tasks with their neural probabilistic language model that jointly learned a representation for each word and learned a probability function for word sequences using these representations, as a solution for the curse of dimensionality problem encountered when modeling language with longer histories. The architecture implemented for this joint learning task has been shown in figure 5.

In this model, input words from a sentence history were first converted to fixed dimensional embeddings, which were then concatenated and provided to a hidden layer and optionally, also provided to the output layer. The hidden layer was connected to the output layer, whose units rep-

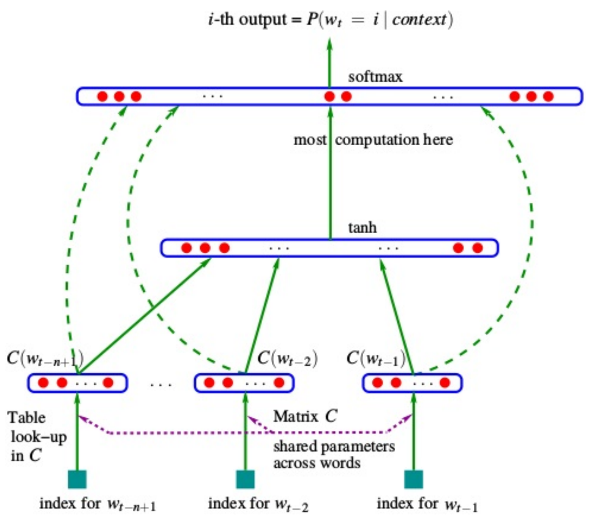


Figure 5: Architecture used by Bengio et al. (2003)

resented words in the vocabulary and a softmax function on the output layer provided a probabilistic output over the next word, given previous words. The network showed significant improvements in the quality of language modeling as compared to other standard models such as Kneser-Ney back-off and class-based back-off. However, while they focused on the learning unique representations for each word, they did not explore the information stored within these 'embeddings'.

Following the success of Bengio et al. (2003), a wide variety of connectionist model-based word embeddings were introduced and studied. Mnih and Hinton (2007) proposed three probabilistic models – a factored restricted Boltzmann machine (RBM) LM, a temporal factored RBM LM, and a log-bilinear LM – for next-word prediction task by using distributed representations of words. The temporal FRBM and the log-bilinear models showed exceptional performance, especially with larger context sizes.

Collobert and Weston (2008) made use of a "lookup table" layer that was essentially a distributed word representation layer that can be trained through backpropagation. Their architecture was designed to be trained jointly for several different NLP tasks – part-of-speech tagging, chunking, named-entity recognition, semantic role labeling, and language modeling – while sharing weights across these tasks.

Mnih and Hinton (2009) introduced a hierarchical log-bilinear model (HLBL) that made use of hierarchical clustering of words. While hierarchical clustering of words using binary trees has been performed before this work, the authors here used a pure learning approach. This is beneficial because one no longer needs expert knowledge to generate such trees. A simple method to achieve this is to use large count vectors to describe the words, but this is problematic because of the exponentially large vocabularies encountered when training on large dataset and with larger context sizes. This also leads to a data sparsity problem that plagued n-gram models as well. To avoid such issues, the authors used low-dimensional real-valued word vectors to build the trees. In this method, the learning of the word representations happens in the tree-building process. A drawback of their method, however, was that it was not able to learn multiple codes for words with multiple meanings. Instead, the tree seemed to replicate infrequent words that were hard to cluster.

A study by Turian et al. (2010) was one of the first to systematically analyze the usefulness of word representations in NLP tasks. The authors sought to compare the performance of near state-of-the-art supervised baseline models augmented by three different word representations – Brown clustering (a class-based bigram LM) (Brown et al., 1992), Collobert and Weston (2008) embeddings, and HLBL embeddings (Mnih and Hinton, 2009). They found all three word representations improved the performance of the baseline models. In their experiments, they also found that word embeddings learned in a semi-supervised task-specific manner performed better than those learned in an unsupervised task-inspecific manner. However the unsupervised method results in word embeddings that can be shared across research studies without need for further training. The authors also point out a drawback of connectionist model-based word embeddings as compared to something like Brown clustering. The Brown clustering is able to better represent rare words, whereas Collobert and Weston (2008) embeddings cannot do so since they do not receive many training updates during backpropagation.

Mikolov et al. (2013a) introduced two new neural network-based models to generate word embeddings (now popular as `word2vec` embeddings). The first model is the continuous bag-of-words (CBOW) model which tries to predict a middle word given N words before and after it in a sequence. This model simply averages the word vectors for the context words and then feeds them to a log-linear classifier to predict the middle word. The other model, called the continuous Skip-gram model, achieves the opposite of the CBOW model. Given a middle word, it tries to predict N words before and after it. The embeddings generated were found to have properties beyond simple word similarity, which we'll discuss in the next section. Given the huge performance improvements gained by these embeddings, they marked a dramatic change in the attitude of the wider research community regarding distributed word representations.

Mikolov et al. (2013b) introduced negative sampling as an alternative to hierarchical softmax as used in their Skip-gram model. This is shown to learn accurate representations for frequent words. In this work they also discussed how simple linear operations on the word vectors lead to other meaningful vectors, and can also be used to represent groups of words like phrases and sentences – more on these in the next couple of sections.

Generating word embeddings across multiple languages was studied by Al-Rfou et al. (2013). Their findings indicate that it is possible learn embeddings over multiple languages jointly. An interesting result from this work is that these jointly learned embeddings also seem to act as word translations between languages. For example, the five nearest neighbors of the French word "rouge" ("red" in English) are "juane", "rose", "blanc", "orange", and "bleu", while the neighbors for the English word are the same corresponding translations of the colors – "yellow", "pink", "white", "orange", and "blue".

Pennington et al. (2014) analyzed the model properties needed to generate word vectors with linguistic regularities as described by Mikolov et al. (2013c). They propose a new global lobbilinear regression model that combines matrix factorization and local window-based methods to generate Global Vector (GloVe) embeddings. These embeddings are shown to outperform the `word2vec` embeddings from the CBOW and Skip-gram models (Mikolov et al., 2013a) on word analogy, word similarity, and named entity recognition tasks.

Most approaches discussed so-far have a common weakness – they are bad at generating representations for infrequent words in the dataset. This

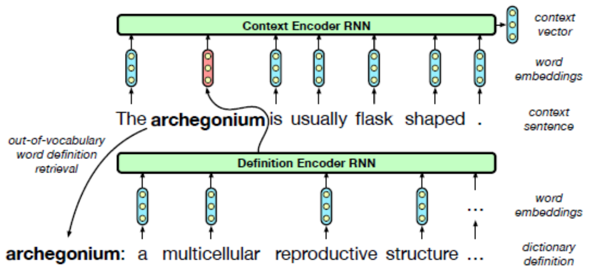

Context Encoder RNN
context vector
word embeddings
The archegonium is usually flask shaped .
context sentence
out-of-vocabulary word definition retrieval
Definition Encoder RNN
word embeddings
archegonium: a multicellular reproductive structure ...
dictionary definition

Figure 6: On-the-fly embedding for out-of-vocabulary word using dictionary definition. (Bahdanau et al., 2017)

is the natural outcome of depending on data – infrequent words will see infrequent updates during backpropagation. To counter this problem, Bahdanau et al. (2017) introduced a method to compute word embeddings on-the-fly using auxiliary data sources. The approach is simple – for infrequent or out-of-vocabulary words, one can generate embeddings using dictionary definitions or relevant Wikipedia articles. An example is provided in figure 6. An advantage of creating embeddings this way can be seen in action in figure 7. This approach can also be used for generating embeddings for groups of words or phrases. Such embeddings can be extremely useful for domain specific texts and bypasses the need for extremely large training sets. This approach also allows to switch the auxiliary sources if dissatisfied with the results from one.

More recently, Peters et al. (2018) described a method to generate embeddings from a bidirectional language model. In this approach, embeddings are extracted from the hidden states of a bidirectional language model – one which is trained using context words from both before and after the current word. This method of generating word embeddings is shown to perform well in capturing different word senses. For example, the word "play" can be both a noun and a verb.

4 Properties of Distributed Word Representations

Distributed representations achieve generalization at a level that n-gram models cannot match and, depending on the method of derivation, they can encapsulate several features of natural language. We have touched upon a few of these properties in previous sections, and will discuss them here in more detail.

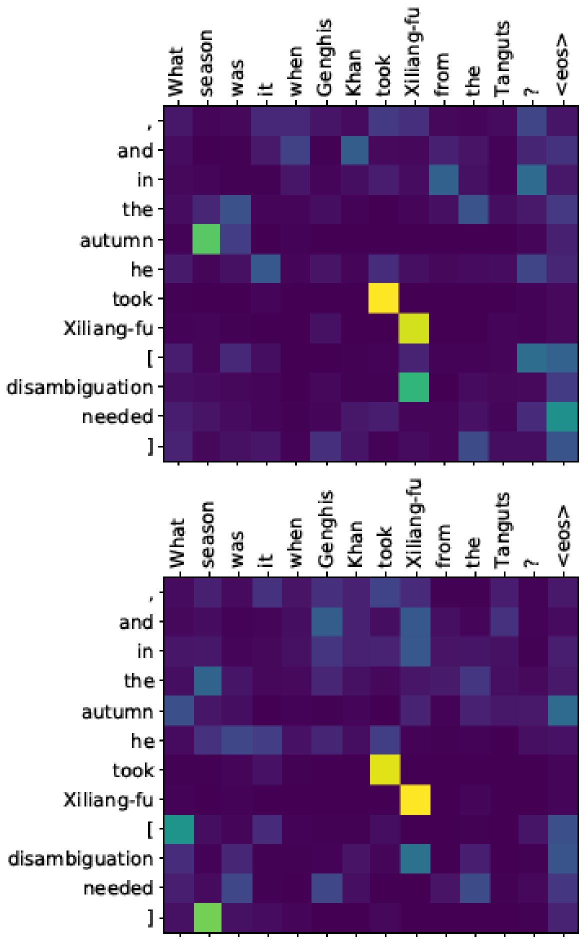


Figure 7: Attention maps of two models – with dictionary embeddings (left) and without (right) – in a question answering task. Note how dictionary embeddings helps in matching "autumn" and "season". (Bahdanau et al., 2017)

4.1 Word Similarity

Word similarity is probably the easiest property to grasp. Similar words will have distributed representation vectors close to each other. In a way, this property exists by definition of distributed word representations. Typically, one can use words with very small scalar differences in their vector representations interchangeably. Related words can also be found near each other. Verifying word similarity is a common test of the quality of word embeddings.

4.2 Word Analogy

Similar to how small scalar differences between vector representations of two words imply semantic similarity, the vector differences between the representations also capture certain information of the words; specifically, the relationship between

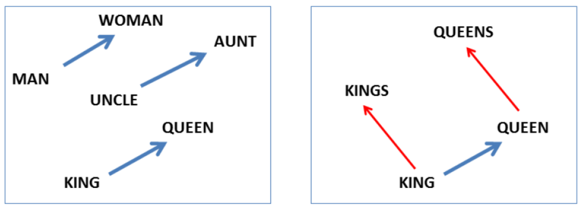


Figure 8: Visual representation of linguistic regularities captured by distributed representations of words. (Mikolov et al., 2013c)

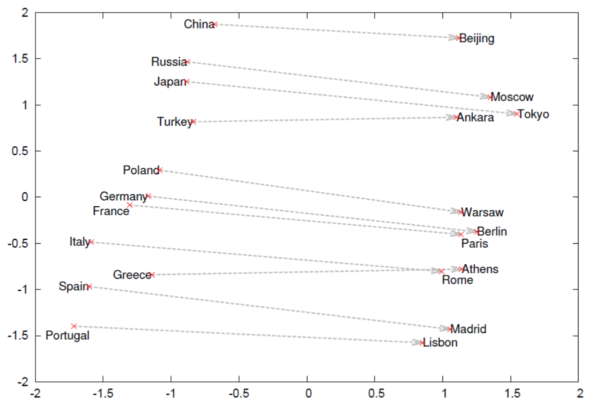


Figure 9: Two-dimensional PCA projection of word vectors of countries and their capital cities. Note the near-uniform vector offset between countries and capitals. (Mikolov et al., 2013b)

the words.

For example, the vector difference between the vectors for "king" and "queen" is nearly equal to that between the vectors for "man" and "woman", and to that between the vectors for "uncle" and "aunt". These regularities can be easily verified by linear operations on the vectors, for example, $vector(\text{“}king\text{”}) - vector(\text{“}man\text{”}) + vector(\text{“}woman\text{”})$ will give a vector that is approximately equal to $vector(\text{“}queen\text{”})$. This relation can be visually understood in figure 8.

Another example, as seen in figure 8, is the vector difference between the vectors for the singular and plural forms of a word. The vector difference between "king" and "kings" should be approximately equal to that between "queen" and "queens".

Mikolov et al. (2013b) also present similar relationships that were captured between countries and their capital cities in their 1000-dimensional word vectors generated by their Skip-gram model. Figure 9 shows the relationship using the top two dimensions as projected by PCA.

Mikolov et al. (2013c) introduced a new evaluation metric for distributed representations based on the vector differences of word vectors. They posit that the quality of a learned distributed representation can be measured by the consistency with which one can retrieve the vector for a word using linear operations on vectors for other words. The metric itself is simply a cosine similarity between the word vector estimated through linear vector operations and the true word vector. Scoring well on this evaluation metric indicates that the dimensions of the learned distributed representations have some meaning.

5 Distributed Representations of Phrases, Sentences, and Documents

So far, we have discussed the distributed representations of words. While being incredibly useful, these can fall short when groups of words in a specific order can mean different things than the individual words themselves. For example, "Air" and "Canada" are two words present in "Air Canada", but the individual words do not represent the airlines Air Canada. For this reason, it is useful to consider distributed representations of phrases, sentences, and even documents.

One of the earliest representations of documents and queries can be found in (Gallant et al., 1992), where the authors make use of learned word context vectors to obtain document representations, by taking a weighted vector sum of context vectors of all the words contained in the document. They obtain representations for phrases in the same manner as documents, by taking weighted vector sum of all word context vectors present in the phrase. While this technique provides a way to represent documents and arbitrary length phrases, it also maps different entities to the same semantic space. This is a potential drawback, since the fact that phrases and documents are composed of words in a given sequence which hold syntactic information has been completely ignored.

Mikolov et al. (2013b) present a simple method for finding phrases. Essentially, they rely on co-occurrence of words, especially when a certain co-occurrence or ordering is more frequent than other occurrences of the individual words. Once phrases are identified, they are treated like a single word and vectors are generated for them as well. This improves the results on specific domain related texts when co-occurrence of words is common and has special meaning attached to it.

Le and Mikolov (2014) introduced a Paragraph Vector framework that includes vectors of sentences, paragraphs, and documents, to improve performance of models that use only word-level representations. The paragraph vectors are able to capture the big-picture concepts of the text, and this mitigates some of the weaknesses of the word-order independent bag-of-words type of models that consider only the word-level representations. Paragraph vectors can be used in a variety of tasks. For example, in the language modeling task of predicting the next word given a set of previous words, including a paragraph vector along with the given history words can add some context to the model.

As discussed in an earlier section, the on-the-fly embedding method from auxiliary data by Bahdanau et al. (2017) can also be used to for generating embeddings for phrases and sentences. One can benefit by using them for named entities with multiple words like "New York Times" and "Coronation Street".

6 Conclusion

In this survey, we presented the most popular techniques of obtaining word representation and showed how these techniques have evolved with time. While the choice of word embeddings for any given NLP task is an open question, it is beyond doubt that distributed representations of words, especially those learned in an unsupervised manner, have the ability to encapsulate several semantic and syntactic features of natural language. Similar or related words can be found near each other in the embedding space, and vector differences between word embeddings can be used in simple linear vector operations to obtain meaningful solutions to tasks like word analogy. The improvements brought on by such representations to existing models are immediate and significant. Motivated by this impact, many researchers are studying efficient methods to generate distributed representations for words, and even characters, phrases, sentences, and documents. While having a rich and long history, as covered in this survey, there are likely to be many more interesting discoveries related to distributed representations awaiting us in the future.

References

Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed word representations for multilingual nlp. *arXiv preprint arXiv:1307.1662* .

Dzmitry Bahdanau, Tom Bosc, Stanisław Jastrzębski, Edward Grefenstette, Pascal Vincent, and Yoshua Bengio. 2017. Learning to compute word embeddings on the fly. *arXiv preprint arXiv:1706.00286* .

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *J. Mach. Learn. Res.* 3:1137–1155. http://dl.acm.org/citation.cfm?id=944919.944966.

John Blitzer, Fernando Pereira, Kilian Q Weinberger, and Lawrence K. Saul. 2005. Hierarchical distributed representations for statistical language modeling. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, MIT Press, pages 185–192. http://papers.nips.cc/paper/2691-hierarchical-distributed-representations-for-statistical-language-modeling.pdf.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics* 18(4):467–479.

William Caid, Susan T. Dumais, and Stephen I. Gallant. 1995. Learned vector-space models for document retrieval.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*. ACM, pages 160–167.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science* 41(6):391.

Jeffrey L. Elman. 1991. Distributed representations, simple recurrent networks, and grammatical structure. *Mach. Learn.* 7(2-3):195–225. https://doi.org/10.1007/BF00114844.

S. I. Gallant. 1991. A practical approach for representing context and for performing word sense disambiguation using neural networks. In *IJCNN-91-Seattle International Joint Conference on Neural Networks*. volume ii, pages 1007 vol.2–. https://doi.org/10.1109/IJCNN.1991.155584.

Stephen I. Gallant, William R. Caid, Joel Carleton, Robert Hecht-Nielsen, Kent Pu Qing, and David Sudbeck. 1992. Hnc's matchplus system. *SIGIR Forum* 26(2):34–38. https://doi.org/10.1145/146565.146569.

G. E. Hinton, J. L. McClelland, and D. E. Rumelhart. 1986. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. MIT Press, Cambridge, MA, USA, chapter Distributed Representations, pages 77–109. http://dl.acm.org/citation.cfm?id=104279.104287.

Geoffrey E. Hinton. 1990. Mapping part-whole hierarchies into connectionist networks. *Artif. Intell.* 46(1-2):47–75. https://doi.org/10.1016/0004-3702(90)90004-J.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International Conference on Machine Learning*. pages 1188–1196.

Kevin Lund and Curt Burgess. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers* 28(2):203–208. https://doi.org/10.3758/BF03204766.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. pages 3111–3119.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013c. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. pages 746–751.

Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*. ACM, pages 641–648.

Andriy Mnih and Geoffrey E Hinton. 2009. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*. pages 1081–1088.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365* .

Yonggang Qiu and Hans-Peter Frei. 1993. Concept based query expansion. In *Proceedings* *of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, USA, SIGIR '93, pages 160–169. https://doi.org/10.1145/160688.160713.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323(6088):533–536. http://dx.doi.org/10.1038/323533a0.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1988. Neurocomputing: Foundations of research. MIT Press, Cambridge, MA, USA, chapter Learning Representations by Back-propagating Errors, pages 696–699. http://dl.acm.org/citation.cfm?id=65669.104451.

Magnus Sahlgren. 2001. Vector-based semantic analysis: Representing word meanings based on random labels. In *In ESSLI Workshop on Semantic Knowledge Acquistion and Categorization*. Kluwer Academic Publishers.

Hinrich Schütze. 1993. Word space. In *Advances in neural information processing systems*. pages 895–902.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, pages 384–394.

David L. Waltz and Jordon B. Pollack. 1988. Connectionist models and their implications: Readings from cognitive science. Ablex Publishing Corp., Norwood, NJ, USA, chapter Massively Parallel Parsing: A Strongly Interactive Model of Natural Language Interpretation, pages 181–204. http://dl.acm.org/citation.cfm?id=54455.54462.