Text Representation Models Based on Neural Networks

Karlo Babić
Department of Informatics,
University of Rijeka,
Radmile Matejčić 2, 51000 Rijeka, Croatia
Email: karlo.babic@inf.uniri.hr

***Abstract*—In natural language processing text has to be transformed into machine-readable representation before any processing on it can be done. The quality of any further NLP tasks greatly depends on the quality of those representations. This work gives an overview of text representation models that use neural networks. For this overview, almost 40 models in research articles from the last decade are analyzed. The models described in those articles are categorized by the representation level as subword, word, and larger parts of text (e.g., phrase, sentence) representations, and by the architecture of neural networks as shallow, convolutional, recurrent, and recursive models (with additional attention and multimodal models). Different representation models are suitable for different NLP tasks. In preliminary work, we tested shallow models on an analogy task and for measuring semantic similarity.**

Keywords - representation, embedding, neural networks, deep learning, NLP

I. Introduction

Text is an important source of human knowledge and a medium through which humans communicate. For artificial intelligence in general, it is very useful to give computers the ability to understand language.

To do so, text first has to be transformed into text representation (represented as vectors of numbers). It is desirable that text representations are encoded in a vector space that preserves information about semantics, syntax, knowledge, and other information that could be useful for downstream tasks (tasks that use text representation). Such a representation would be closer to the human way of perceiving text. Text representation is directly needed in tasks such as information retrieval, question answering, textual entailment identification, keyword extraction, text summarization, sentiment analysis, machine translation, etc.

Generally, today there exist numerous approaches for solving various NLP tasks. Some early approaches performed text processing by using a set of rules [1]. While later in the 1980s, there was a revolution in the field of NLP with the introduction of machine learning algorithms and statistical approaches. Probabilistic language models (e.g., n-grams) assign probabilities to sequences of words [2]. In the last decade, the most successful approaches include representation learning based on neural network models [3].

This overview is focused only on models based on neural networks that produce vector representations in n-dimensional space. Each vector in that space represents a text element with the desired propriety that the vector is closer to all other vectors that are similar to it. This similarity usually refers to the semantic similarity. Except for the closeness based on semantics, the vector space may be meaningfully structured in many other ways, capturing syntax, knowledge, and other information. Text elements for which representations can be created are subwords, words, phrases, sentences, and documents. The process of representing text elements as vectors is called embedding (e.g., word embedding).

The simplest representation would be if each word gets its own number representation, but in such a representation all words would be represented in one dimension. A one-dimensional representation can not preserve useful information about the words (such as semantic information and syntax).

One-dimensional problem is solved by one-hot encoding, where every word gets its own dimension, and the length of the representation vector is equal to the number of words in the vocabulary. One-hot representation has for a word a ”0” for each element in the vector except for one element, which is ”1”. In this representation, each word is equally distanced from every other word, which still does not preserve information about language and meaning, but at least this representation does not preserve irrelevant information about distances between words as the first method, which is one dimensional. One more problem with one-hot encoding is that the representation vectors are very long.

Early representation methods for texts (documents, paragraphs, or sentences) used the bag-of-words approach, which preserves only the information about which words are in the text. Better methods use tf-idf (term frequency - inverse document frequency), LSA (latent semantic analysis), and similar models that hold more information as they preserve some statistics about the words in the text [4].

One-hot and tf-idf based methods are sparse vector methods (as the vector that represents a text has zeros for all the words that are not in the text it represents). New methods, which produce dense vectors, use techniques that generate representations that preserve more information about the text. Those techniques are more promising not only because they can potentially preserve dense information, but because the produced vectors are smaller as well (which makes training downstream tasks easier).

As is mentioned in the book [5]: generally, a good representation is one that makes a downstream learning task easier. Representation learning is by its nature unsupervised learning problem, which can be very useful with the amount of unlabeled data we now have. Representation learning can be used as a pretraining method where a neural network, which would use that pretrained representation, could learn faster and perform better on another task. Representations can be used for transfer learning as well, where representation learned for one task can be useful for other tasks.

Text representation is needed for a wide variety of tasks, and neural models appear to generate high-quality representations. Because of the nature of neural networks, when a model has learned to perform well at a task, a byproduct is a dense representation formed in the hidden layers of the neural network.

The aim of this research is to identify what are the promising directions in the research of neural text representations models. To do so, we analyze almost 40 models from the last decade published in research papers in prominent journals and NLP conferences. Models are categorized by the representation level and architecture of neural networks.

Amongst similar overviews that analyzed neural network models for text representation are a survey on neural language models [6] and a bachelor's thesis that reviewed word embedding models [7]. Authors in [6] covered neural network language model methods, which do produce text representation. Authors in [7] covered word representation models (and document similarity algorithms). Our overview reviews more neural network architectures that are used, and all the text representation levels.

The rest of the paper is organized as follows. Section 2 covers related work, Section 3 overviews representation methods, Section 4 covers preliminary results, Section 5 shows the possible future work, and Section 6 concludes the paper.

II. Related Work

The model that popularised the usage of shallow neural networks for word representations is Word2Vec [8, 9]. It uses a fully connected neural network (*FCNN*) **shallow** architecture that learns statistics about word contexts (Figure 1). It follows the distributional hypothesis, which states that the words which are similar in meaning occur in similar contexts [10]. In the learned space, two word representations are closer together for the two words that are more similar, and the relationships between words are preserved as well (where king and man are positioned in the same way as queen and woman, and the positions of countries and their capitals preserve meaningful information as is shown in Figure 2).

Convolutional neural networks (*CNN*) work successfully with visual data, they are used less frequently for text representation (architecture shown in Figure 3). One early work [12] used CNNs to train word representations by learning to predict if the word in the middle of the input window is related to its context or not.

Words in a text are following some meaningful order. Models like Word2Vec are not sensitive to word order, they

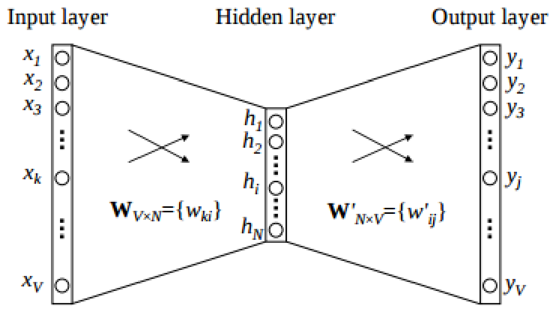

Fig. 1. Word2Vec (shallow) arhitecture [11].

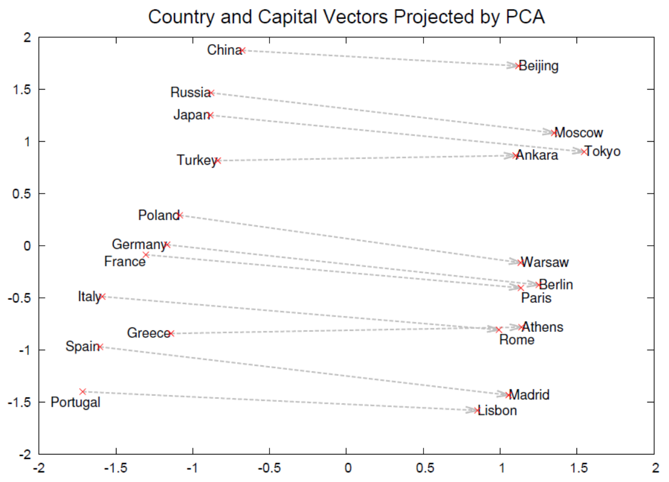

Fig. 2. Country and capital vector representations [8].

are only interested if a word is in the context of another word. **Recurrent neural network** (*RNN*) models create text representations with recurrent units, which besides sending signals forward, send signals back to themselves. By doing so, recurrent layers can learn representations of sequences by receiving input tokens one by one while the recurrent units preserve previous states. Such models are sensitive to word order. One example is Seq2Seq [14] that uses long short-term memory (*LSTM*) layers, a type of RNN layer that solves some of the RNN problems (most importantly vanishing gradient).

Recursive neural network (*RvNN*) models are sensitive not only to word order but to the structure of a text as well (example of recursive neural network predicting word sentiment classes is shown in Figure 5). One of the more successful RvNN models is RNTN (Recursive Neural Tensor Network) [16]. RNTN learns text representations for sentiment classification. Because of the nature of RvNN models, while learning text representations with the help of parse trees (which can be given to the model, learned by the model, or generated by the model) model learns not only sentence representations, but word representations, and phrase representations for every node in a parse tree.

Inspired by how humans read and understand longer texts, **neural attention mechanism** was created. When processing longer texts, an RNN would not work very well because it

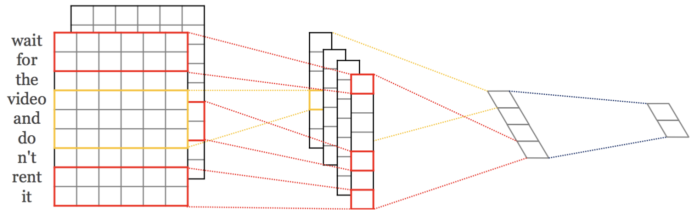

Fig. 3. Convolutional arhitecture [13].

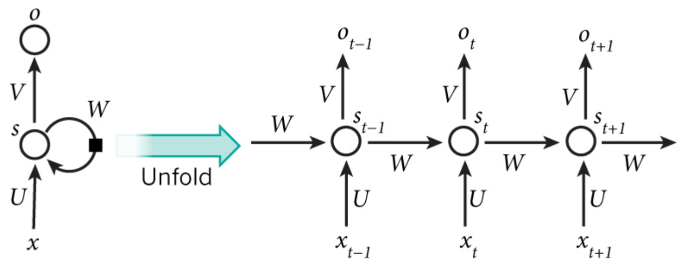

Fig. 4. Recurrent architecture, and the unfolding in time [15].

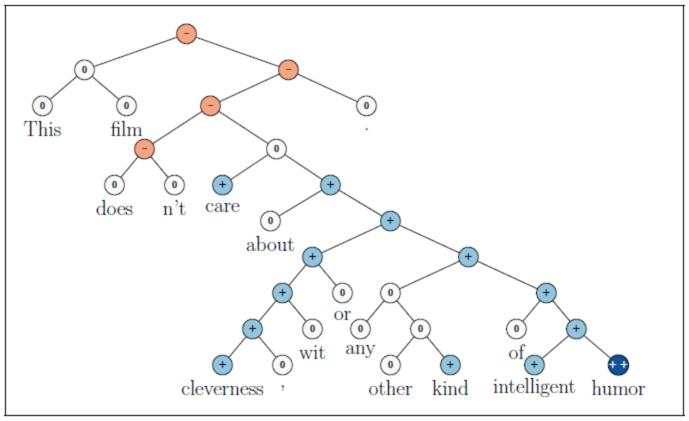

Fig. 5. Recursive neural network predicting word sentiment classes [16].

focuses equally on every word in a text for every decision. Neural networks with attention can focus on parts of a text that are more important for a current task, and thus work better with long texts. Currently, the most promising method that is using the attention mechanism is BERT (Bidirectional Encoder Representations from Transformers) [17]. BERT transformer uses a bidirectional self-attention mechanism. While learning text representation, it masks a percentage of input words and tries to predict them.

Humans learn concepts not only by reading about them, but we also have other sources of information as well (vision, sound, previous knowledge, etc.). Some methods (as [18]) use images besides text to improve text representation. As the model learns from multiple sources of information, and the information is grounded to some additional knowledge while learning, this kind of learning is called **multimodal** or **grounded** learning.

III. Overview

A. Representation Levels

Text can be represented on multiple levels. Models can learn representations for subwords, words, phrases, sentences, paragraphs, or documents.

Table I categorises some of the neural network models into subword, word, and sentence+ representation levels.

Subword elements include characters and character n-grams. Downstream tasks in languages with rich morphologies perform better with subword representations, and out-of-vocabulary words are easily represented [19].

Word representations are easier to implement, as generated representations are ready for use, while subword representations have to be combined into word representations.

Sentence+ is a category for phrase, sentence, paragraph, and document representations. While word representations can be combined to form a sentence+ representation, neural network models that are specialized for learning sentence+ representations learn better representation spaces.

TABLE I
Categorisation of NN models by representation levels

Level	Model
subword	CharSCNN [20], CharWNN [21], CharCNN [22], FastText [23], GPT-2 [24]
word	[12], Word2Vec [8, 9], [25], Deps [26], GloVe [27], [18], [28], ELMo [29], ConvRR [30]
sentence+	[31], [32], RAE [33], MV-RNN [34], cimRNN and csmRNN [35], RNTN [16], [36], Seq2Seq [14], Doc2Vec [37], [38], Skip-Thoughts [39], RCNN [40], Tree-LSTM [41], TBCNN [42], SPINN [43], RL-SPINN [44], ST-Gumbel [45], [46], [47], [46], GLoMo [48], BERT [17]

B. Neural Network Architectures

Neural network models that learn text representation most frequently use shallow, convolutional, recurrent, recursive, or a combination of mentioned architectures.

Table II categorizes some of the neural network models into shallow, convolutional, recurrent, and recursive architectures.

Shallow architectures in this categorization mainly include feed-forward neural networks that have one hidden layer. Such models learn only a very shallow representation of text. They perform well on simple tasks like measuring word similarity.

Convolutional architectures are well suited for learning local patterns in text. Pooling layers learn to detect important tokens or features for a specific task it is trained on [3].

Recurrent architectures are made for sequential data. Text input can be treated like sequential data, reading input text token by token while updating network states.

Recursive architectures read texts in a structured fashion (parsing trees). Because of that, recursive models create high-quality sentence representations and perform very good in semantic and sentiment tasks [41].

TABLE II
CATEGORISATION OF NN MODELS BY ARCHITECTURE

Architecture	Model
shallow	Word2Vec [8, 9], Deps [26], GloVe [27], Doc2Vec [37], FastText [23]
convolutional	[12], CharSCNN [20], CharWNN [21], [28], [22] (CharCNN), [49], ConvRR [30]
recurrent	[38], Seq2Seq [14], Skip-Thoughts [39], RCNN [40], [46], ELMo [29], [47], GLoMo [48]
recursive	[31], [32], RAE [33], MV-RNN [34], cim-RNN and csmRNN [35], RNTN [16], [36], Tree-LSTM [41], TBCNN [42], SPINN [43], RL-SPINN [44], ST-Gumbel [45]

C. Shallow Models

In shallow models phrase, sentence, and document representations are made by combining shallow word representations, which makes the resultant representations very simplistic (bag of words). Shallow models learn a very shallow text representation and therefore are very fast to train, and can be used as input for models that can create better representations.

Shallow models perform well only on simple tasks (e.g., word similarity). If a task is more complex (requires a deeper understanding, e.g., question answering, summarization) than shallow models are not the right choice.

Word2Vec [8, 9] popularised shallow word representation using neural networks. Word2Vec has two models: continuous bag-of-words (*CBOW*) and skip-gram. Both learn word representations through unsupervised learning. The CBOW model scans over the text with a context window around the target word and it learns to predict the target word from the context words. The skip-gram model learns to predict the context words from the target word. Word2Vec neural network has only one hidden layer, and word representations are extracted from that layer.

Deps [26] exploits dependency parse trees. Deps model generalizes skip-gram to include arbitrary contexts and uses dependency-based contexts derived from parse trees.

GloVe [27] directly captures global corpus statistics through unsupervised learning. GloVe combines global matrix factorization and local context window methods through a bilinear regression model. By doing so it learns from the co-occurrence matrix and trains word representations in a way that it can predict co-occurrence ratios. An example was given in [27]. Let $i = ice$ and $j = steam$, if $k = solid$, we expect the ratio P_{ik}/P_{jk} to be large (P_{xy} is probability of words x and y to occur together). If $k = gas$, the ratio should be small. For words that are related to both ice and $steam$, or to neither, the ratio should be closer to 1. This is used instead of raw probabilities because with ratios we can more easily distinguish relevant words from irrelevant words. Mathematically GloVe is similar to Word2Vec [50].

Doc2Vec [37] is an extension of Word2Vec that can learn representations for documents (or smaller parts of texts). While predicting context words of a target word, Doc2Vec receives on input the target word and the document ID. Through learning, it learns not only representations for words, but documents as well.

FastText [23] is an unsupervised model that learns representations for character n-grams, and each word is represented as a bag-of-character n-grams. Previous models were limited to assigning a distinct vector to each word. Representations on the subword level are shown to be better for morphologically richer languages and for rare words.

D. Convolutional Models

Convolutional architectures are widely used for computer vision tasks. CNNs by their nature learn to abstract input data through multiple levels and detect patterns on each level.

The model introduced in **[12]** used a single convolutional neural network architecture that, given a sentence on input, outputs part-of-speech tags, chunks, named entity tags, semantic roles, semantically similar words, and the likelihood that the sentence makes sense. All of these tasks are trained jointly. The language model is learned through unsupervised learning, while the other tasks are learned through supervised learning. While learning, all the tasks share weights. When the neural network is trained, word representations can be extracted from it.

CharSCNN [20] (Character to Sentence Convolutional Neural Network) is a supervised deep convolutional neural network that exploits from character-level to sentence-level information to perform sentiment analysis of texts. Representations can be extracted from each level: character, word, and sentence.

CharWNN [21] is a supervised model that is trained on part-of-speech tags. It learns character-level representations that can then be used to detect word morphology.

The model proposed in **[28]** is a supervised triplet network model, which aims to learn useful representations by distance comparisons. The network learns word representations by predicting which two of the three words received on input are in the same class.

CharCNN [22] (character-level Convolutional Neural Network) is an unsupervised model that relies only on character-level inputs. It uses CNN and LSTM. Except for the character representations, word representations can be extracted from the network as well.

ConvRR [30] (convolutional Residual Retrieval network) proposed a new unsupervised method (based on CNN) to generate multi-resolution word representations. Multi-resolution means that multiple word representations are combined in the network to take advantage of the strengths of each of the word embedding methods.

E. Recurrent Models

Recurrent neural networks process input as a sequence and learned representations preserve ordered knowledge about a text. Such networks are well suited for sentences or phrases. While tokens from the input sequence are processed, the history of all the previous tokens is preserved as a state in the neurons. Normal RNNs (recurrent neural network) have a vanishing (and exploding) gradient problem where with longer texts the network forgets about the older inputs [51]. The vanishing gradient problem is solved by GRUs (gated recurrent unit) and LSTMs (long short term memory).

A supervised encoder-decoder model in **[38]** uses two RNNs. One network encodes a sequence of symbols into a representation vector, and the other decodes the representation into another sequence of symbols. The sequence of symbols can be a phrase or a sentence.

Seq2Seq [14] is a supervised recurrent neural network model. More specifically, it uses LSTM units in recurrent layers. Seq2Seq transforms the input sequence into the output sequence, and it is trained on a machine translation task. This model can produce representations for phrases and sentences. With the help of LSTM units, this network performs better on longer sequences than RNN models.

Skip-Thoughts [39] is an unsupervised encoder-decoder model that learns to reconstruct the surrounding sentences of an encoded sentence. They use encoder with GRU activations and an RNN decoder with a conditional GRU. Skip-Thoughts learns representations for sentences and performs equally as the LSTM approach.

RCNN [40] (Recurrent Convolutional Neural Network) is a supervised bidirectional RNN (*BiRNN*) with a pooling layer after the BiRNN layer. Sentence representations are learned through the BiRNN, which scans over texts in both directions, while normal RNNs scan texts only in one direction. The pooling layer learns to select the most important words for a text classification task.

ELMo [29] (Embeddings from Language Models) is an unsupervised model that uses bidirectional LSTM (*BiLSTM*). This model solves a word representation problem where in previous models (e.g. Word2Vec) each word has one representation vector, but each word can have a different meaning in different contexts (polysemy). ELMos word representations are learned functions of the internal states of the BiLSTM. In different contexts, the same word has different representation vectors.

A semi-supervised model in **[47]** uses bidirectional GRU (*BiGRU*) for sentence representation multi-task learning. Tasks that the model is trained on are skip-thought, machine translation, constituency parsing, and natural language inference.

GLoMo [48] (Graphs from Low-level unit Modeling) is an unsupervised model with a complex architecture for learning generic latent relational graphs that captures dependencies between pairs of data units (e.g. words). Those graph representations are used as sentence representations. The model architecture consists of RNN, CNN, and Attention.

F. Recursive Models

Recurrent models learn representations with ordered information, but recursive neural networks go one step further and learn structured information. Recursive networks process inputs in recursive fashion through a tree structure. Each node in that tree structure has associated with it a representation. Those trees can be given on input, learned from annotated texts, or generated by a neural network without learning tree structures (latent trees).

A supervised model in **[31]** learns to parse sentences (create parse trees). While learning how to parse sentences, this model learns text representation as well.

Authors introduced in **[32]** a supervised structure prediction model that can recover recursive structures in the inputs of natural scene images or natural language sentences. While learning text structure, it learns text representation.

RAE [33] (Recursive Autoencoders) introduced an architecture based on recursive autoencoders for sentence-level prediction of sentiment label distributions. While this model can learn text representation through supervised learning of sentiment distributions, it can learn text representation through unsupervised learning as well. Parsing trees are latent, meaning that they are not learned or given on input, but generated by concatenating neighboring pairs of words or phrases and combining the ones with the lowest reconstruction error (in autoencoder) into parent nodes.

MV-RNN [34] (Recursive Matrix-Vector Model) introduced a model that learns vector and matrix representations for every node in the tree (phrases and sentences). The vector captures the meaning of the node, while the matrix captures how it changes the meaning of neighboring words or phrases. Supervised training of the model is done by adding on top of each parent node a softmax classifier. The model uses given parse trees.

Some other supervised recursive models that work on given parse trees are RNTN [16], [36], Tree-LSTM [41] and TBCNN [42].

Models **cimRNN** and **csmRNN [35]** learn word representations through unsupervised training with the help of morpheme trees, while all the other recursive neural networks in this overview learn sentence+ representations.

Tree-LSTM [41] introduced a generalisation of LSTM to tree-structured network topologies. The model is trained on a sentiment task.

SPINN [43] (Stack augmented Parser-Interpreter Neural Network) introduced a model architecture (SPINN-PI-NT) that is equivalent to the Tree-LSTM model. Text representation and tree structures are learned through supervised training.

RL-SPINN [44] uses reinforcement learning to learn sentence representations. RL-SPINNs architecture is based on the SPINN model. Parsing trees it uses are latent, meaning that the tree structures are not learned or given on input, but are optimized for the downstream task.

ST-Gumbel [45] (Straight-Through Gumbel-Softmax estimator) is based on the Tree-LSTM model. It uses latent trees that are computed with a composition query vector that measures the validity of a composition. Text representation is learned through unsupervised training.

G. Attention Models

Most of the sequence translation models use recurrent or convolutional neural networks with an encoder and a decoder. Some also connect the encoder and decoder through an attention mechanism. Transformer [52] is a neural network architecture based only on attention mechanisms, without using recurrent or convolutional layers.

GPT-2 [24] (Generative Pretrained Transformer 2) is a Transformer that through unsupervised training learns byte sequence representations. The main task of this model is language modeling.

BERT [17] (Bidirectional Encoder Representations from Transformers) is deep bidirectional Transformer. Deep bidirectional means that it is conditioned on every word in the left and right contexts at the same time. It does so by masking some percentage of the input tokens at random and then predicts those masked tokens. BERT is an unsupervised model (language model) that learns sentence representations.

H. Multimodal Models

Multimodal models learn from at least two different sources of information, in models mentioned bellow, those sources are text and images. This is done to create text representations that are more "knowledgeable" about the concepts that are associated with the words. For example, if learning text representations with the help of images, those representations can learn the semantics of objects (e.g. by seeing wheels on cars it can create representations that reflect that knowledge).

The model introduced in **[25]** creates word representations by concatenating Word2Vec word representation vectors with image representation vectors extracted from CNN trained on labeled object recognition dataset.

The model introduced in **[18]** uses stacked autoencoders to learn word representations from textual and visual input. Model is semi-supervised even though autoencoders are used, which are by their nature unsupervised. Model is not completely unsupervised because pairs of words and images have to be paired correctly for the model to learn correct meaningful representations.

A supervised model in **[46]** is trained by encoding sentences to predict image features. The model uses BiLSTM and CNN (ResNet).

A model in **[49]** is trained on the task of visual question answering, where questions are asked about images. The model uses CNN for learning text representations.

I. Evaluation Methods

The quality of the learned representation space has to be somehow evaluated. The evaluations are done indirectly on different tasks, and each task can provide one perspective about the representation space quality. Those tasks range from simpler ones to more complex ones.

Word/sentece similarity task is widely used, but is too simple to be able to test better representations (even shallow text representations perform good on such tasks) (Word2Vec [8, 9], Deps [26], GloVe [27], FastText [23], Skip-Thoughts [39], cimRNN and csmRNN [35], RL-SPINN [44]). The most used evaluation measure for the similarity task are Spearman and sometimes Pearson correlations.

Word analogy task can be used as well, and can tell us the representation space structure quality (GloVe [27], FastText [23]). The most used evaluation measure for the word analogy task is accuracy.

Sentiment classification task can provide more information about the sentence representation quality (Doc2Vec [37], CharSCNN [20], RAE [33], MV-RNN [34], RNTN [16], [36], Tree-LSTM [41], TBCNN [42]). The most used evaluation measure for the sentiment classification task is accuracy.

Translation task is able to inform us about the sentence encoding quality ([38], Seq2Seq [14]). The most used evaluation measure for the translation task is BLEU score.

Representations that perform well at **question answering** tasks preserve knowledge and understanding of text semantics (ELMo [29]). The most used evaluation measure for the question answering task is the F1 score.

Inference task is able of high quality evaluation as well (SPINN [43], ST-Gumbel [45]). The most used evaluation measure for the inference task is accuracy.

IV. Preliminary Results

This section describes preliminary results for word representation comparison and sentence similarity methods with word representations (that uses external knowledge). We perform those two experiments with shallow representations.

In the first preliminary experiment, we compare four word representation models on the analogy task (Table III). FastText proved to be very fast to train, and it is performing better than other models. Other models produced worse results because the data we trained all the models on was small, and FastText learned the most out of it because it is trained on the subword level. When training on the subword level, the tokens are repeated trough the training process more frequently, effectively providing the model with more training examples. The differences in results categorized by analogy themes between Word2Vec, GloVe, and ELMo could possibly provide information about deeper differences of the models. ELMo performed very poorly because when producing representation for a word the model should take into consideration the context of that word, but in our evaluation, there were no contexts.

In the second preliminary experiment [53], we explore the results obtained with two publicly available pre-trained word embeddings (one based on Word2Vec trained on a specific dataset and the second extending it with embeddings of word senses). We test five approaches for aggregating words into text (Tables IV and V). Two approaches are based on centroids and summarize a text as a word embedding. The other approaches are some variations of the Okapi BM25 function and provide directly a measure of the similarity of two texts.

To better understand the nature of the training process of neural networks, in previous work we created visualization methods that visualize the change of the weights through the time of the learning process [54].

The conclusion was that for richer text representation better methods have to be used. Methods like Word2Vec are shallow, so more complex methods that use the structure of the text to learn representations are needed to create better representations.

TABLE III
ACCURACIES ON ANALOGY TASK BY CATEGORY FOR EACH METHOD

	Word2Vec	**GloVe**	**FastText**	**ELMo**
capital-common-countries	0%	30.8%	66%	3.4%
capital-world	0%	5.4%	20.4%	1.7%
currency	0.1%	0%	1.7%	0.3%
city-in-state	0%	19%	15.6%	7%
family	57.1%	29.2%	42.5%	23.5%
gram1-adjective-to-adverb	8.7%	0.6%	59.4%	1.8%
gram2-opposite	9.1%	0%	50.2%	3.8%
gram3-comparative	58.1%	6%	64%	8.5%
gram4-superlative	19.3%	1%	49.8%	3.1%
gram5-present-participle	39.8%	8.7%	51.7%	7.8%
gram6-nationality-adjective	0%	17.9%	79.8%	5%
gram7-past-tense	45%	14.2%	43.5%	11.6%
gram8-plural	36.1%	7%	62.5%	1.8%
gram9-plural-verbs	39%	5.3%	64.9%	10.2%

TABLE IV
PEARSON (r) AND SPEARMAN (ρ) CORRELATIONS OF FIVE SIMILARITY MEASURES FOR *Word2Vec* (A1) AND *NASARI+Word2Vec* (A2) APPROACHES FOR THE $SICK$ DATASET.

	r (a1)	ρ (a1)	r (a2)	ρ (a2)
sim_{cos}	0.642	**0.585**	0.586	0.**557**
sim_{cos2}	**0.661**	0.579	0.604	0.550
sts	0.503	0.468	0.470	0.443
sts_s	0.565	0.534	0.549	0.516
sts_{s2}	0.642	0.537	**0.612**	0.520

TABLE V
PEARSON (r) AND SPEARMAN (ρ) CORRELATIONS OF FIVE SIMILARITY MEASURES FOR *Word2Vec* (A1) AND *NASARI+Word2Vec* (A2) APPROACHES FOR THE Lee DATASET.

	r (a1)	ρ (a1)	r (a2)	ρ (a2)
sim_{cos}	0.582	**0.519**	0.472	0.464
sim_{cos2}	**0.589**	**0.519**	**0.502**	**0.478**
sts	0.283	0.193	0.276	0.225
sts_s	0.474	0.293	0.500	0.406
sts_{s2}	0.424	0.288	0.444	0.378

V. FUTURE WORK

Text representation is very important for NLP, and there is much room for improvement. For future work, we could create a new neural text representation model, a new learning method, or improve an existing model or a method.

A. Recursive Text Representation

A recursive neural network that processes text in a tree-structured way seems promising. One approach that could perform well is a recursive neural network that learns multi-word and sentence representations by encoding neighboring pairs of nodes (nodes present words or sequences of words) and predicting words to the left and to the right of that node. The method could be explained as a recursive skip-gram model. RAE [33] is similar, but instead of predicting neighboring words it is encoding and then decoding the input nodes.

The trees in the new model could be latent, by combining the nodes that best predict the neighboring words. That would be a greedy approach, a more precise approach (and more computationally expansive one) would search for tree structures that in total make the best predictions for the neighbors for all the nodes.

When evaluating the new model, the results could be compared with the RAE model and some other state of the art recursive models.

B. Online Representation Learning

Text representation could be improved without changing the neural network architecture of existing models. One possible approach is to modify learning algorithms so that instead of using one-hot vectors or other pretrained text representations as input, the representation being learned currently in the model could be used as the input while the representation is being learned.

In each iteration, new representations for text can be calculated by multiplying the old representation (input vector) and the hidden representation (representation layer). New representations are then used in the next iteration as the input vector. This approach is almost certainly very unstable.

Another approach that is possibly more stable calculates new input representations every n iterations. Giving some time for the neural network to stabilize.

Evaluation is very direct and simple, the performance of the improved model is directly comparable with the base model.

VI. CONCLUSION AND DISCUSSION

There is a lot of approaches to learning text representations to choose from. From shallower models that are good at simple tasks to more complex models that care about order and structure. For text processing, text representation is the first and the most important step. As such the quality of representations has a high influence on the performance on the downstream tasks.

Shallow models are fast to train and perform well enough on simple tasks (e.g., similarity) on subword and word level.

Recurrent models perform well, and in contrast to recursive models, they do not need additional data (trees) to parse sentences. Recursive models are promising, and latent tree generation can bypass the need for additional data.

Recurrent and attention models are well suited for text generation, while recursive models perform very well for sentiment classification. Convolutional models work well with multimodal learning (in the case when the other source of training data is images).

Regarding the representation levels, the conclusion is that subword representations work better with languages with rich morphology, and out-of-vocabulary words are less of a problem. Both recurrent and recursive models can learn subword representations. Word representations are suitable when out-of-vocabulary words are infrequent and the morphology of the language is not complex. For phrase, sentence, and document representations, models that learn sentence+ representation are the best choice.

References

[1] T. Winograd, "Procedures as a representation for data in a computer program for understanding natural language," MASSACHUSETTS INST OF TECH CAMBRIDGE PROJECT MAC, Tech. Rep., 1971.

[2] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[3] Y. Goldberg, "A primer on neural network models for natural language processing," *Journal of Artificial Intelligence Research 57*, 2016.

[4] S. Robertson, "Understanding inverse document frequency: on theoretical arguments for idf," *Journal of documentation*, vol. 60, no. 5, pp. 503–520, 2004.

[5] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[6] K. Jing, J. Xu, and B. He, "A survey on neural network language models," *arXiv preprint arXiv:1906.03591*, 2019.

[7] J. E. Alvarez and H. Bast, "A review of word embedding and document similarity algorithms applied to academic text," 2017.

[8] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[10] H. Rubenstein and J. B. Goodenough, "Contextual correlates of synonymy," *Communications of the ACM*, vol. 8, no. 10, pp. 627–633, 1965.

[11] "Introduction to word embedding and word2vec," https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa, accessed: 2019-10-24.

[12] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 160–167.

[13] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[14] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[15] "Recurrent neural networks tutorial," http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/, accessed: 2019-10-24.

[16] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[18] C. Silberer and M. Lapata, "Learning grounded meaning representations with autoencoders," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2014, pp. 721–732.

[19] J. Botha and P. Blunsom, "Compositional morphology for word representations and language modelling," in *International Conference on Machine Learning*, 2014, pp. 1899–1907.

[20] C. Dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2014, pp. 69–78.

[21] C. N. d. Santos and V. Guimaraes, "Boosting named entity recognition with neural character embeddings," *arXiv preprint arXiv:1505.05008*, 2015.

[22] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, "Character-aware neural language models," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[23] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[24] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, 2019.

[25] D. Kiela and L. Bottou, "Learning image embeddings using convolutional neural networks for improved multi-modal semantics," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 36–45.

[26] O. Levy and Y. Goldberg, "Dependency-based word embeddings," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2014, pp. 302–308.

[27] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[28] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International Workshop on Similarity-Based Pattern Recognition*. Springer, 2015, pp. 84–92.

[29] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[30] T. Cakaloglu and X. Xu, "A multi-resolution word embedding for document retrieval from large unstructured knowledge bases," *arXiv preprint arXiv:1902.00663*, 2019.

[31] R. Socher, C. D. Manning, and A. Y. Ng, "Learning continuous phrase representations and syntactic parsing with recursive neural networks," in *Proceedings of the NIPS-2010 deep learning and unsupervised feature learning workshop*, vol. 2010, 2010, pp. 1–9.

[32] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng, "Parsing natural scenes and natural language with recursive neural networks," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 129–136.

[33] R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning, "Semi-supervised recursive autoencoders for predicting sentiment distributions," in *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics, 2011, pp. 151–161.

[34] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, "Semantic compositionality through recursive matrix-vector spaces," in *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*. Association for Computational Linguistics, 2012, pp. 1201–1211.

[35] T. Luong, R. Socher, and C. Manning, "Better word representations with recursive neural networks for morphology," in *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, 2013, pp. 104–113.

[36] O. Irsoy and C. Cardie, "Deep recursive neural networks for compositionality in language," in *Advances in neural information processing systems*, 2014, pp. 2096–2104.

[37] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*, 2014, pp. 1188–1196.

[38] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[39] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler, "Skip-thought vectors," in *Advances in neural information processing systems*, 2015, pp. 3294–3302.

[40] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification," in *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[41] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," *arXiv preprint arXiv:1503.00075*, 2015.

[42] L. Mou, H. Peng, G. Li, Y. Xu, L. Zhang, and Z. Jin, "Discriminative neural sentence modeling by tree-based convolution," *arXiv preprint arXiv:1504.01106*, 2015.

[43] S. R. Bowman, J. Gauthier, A. Rastogi, R. Gupta, C. D. Manning, and C. Potts, "A fast unified model for parsing and sentence understanding," *arXiv preprint arXiv:1603.06021*, 2016.

[44] D. Yogatama, P. Blunsom, C. Dyer, E. Grefenstette, and W. Ling, "Learning to compose words into sentences with reinforcement learning," *arXiv preprint arXiv:1611.09100*, 2016.

[45] J. Choi, K. M. Yoo, and S.-g. Lee, "Learning to compose task-specific tree structures," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[46] D. Kiela, A. Conneau, A. Jabri, and M. Nickel, "Learning visually grounded sentence representations," *arXiv preprint arXiv:1707.06320*, 2017.

[47] S. Subramanian, A. Trischler, Y. Bengio, and C. J. Pal, "Learning general purpose distributed sentence representations via large scale multi-task learning," *arXiv preprint arXiv:1804.00079*, 2018.

[48] Z. Yang, J. Zhao, B. Dhingra, K. He, W. W. Cohen, R. Salakhutdinov, and Y. LeCun, "Glomo: Unsupervisedly learned relational graphs as transferable representations," *arXiv preprint arXiv:1806.05662*, 2018.

[49] Z. Wang and S. Ji, "Learning convolutional text representations for visual question answering," in *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 2018, pp. 594–602.

[50] T. Shi and Z. Liu, "Linking glove with word2vec," *arXiv preprint arXiv:1411.5595*, 2014.

[51] Y. Bengio, P. Simard, P. Frasconi *et al.*, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[52] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[53] K. Babić, S. Martinčić-Ipšić, A. Meštrović, and F. Guerra, "Short texts semantic similarity based on word embeddings," in *2019 30th International Scientific Conference on Information and Intelligent Systems (CECIIS)*. FOI, 2019, pp. 27–34.

[54] K. Babić and A. Meštrović, "Visualizations of the training process of neural networks," in *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, 2019, pp. 1619–1623.