Recent Trends in Named Entity Recognition (NER)

Arya Roy ARYAR@ALUMNI.CMU.EDU
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Abstract

The availability of large amounts of computer-readable textual data and hardware that can process the data has shifted the focus of knowledge projects towards deep learning architecture. Natural Language Processing, particularly the task of Named Entity Recognition is no exception. The bulk of the learning methods that have produced state-of-the-art results have changed the deep learning model, the training method used, the training data itself or the encoding of the output of the NER system. In this paper, we review significant learning methods that have been employed for NER in the recent past and how they came about from the linear learning methods of the past. We also cover the progress of related tasks that are upstream or downstream to NER eg. sequence tagging, entity linking etc. wherever the processes in question have also improved NER results.

Keywords: named entity recognition, sequence tagging, deep learning, conditional random fields, bidirectional long short term memory, recurrent neural network, word embedding

1. Introduction

'Named Entity Recognition' refers to the Natural Language Processing task of identifying important objects (eg. person, organization, location) from text. Henceforth we will use NER to refer to Named Entity Recognition and Classification. NER belongs to a general class of problems in NLP called sequence tagging (Erdogan, 2010). Sequence Tagging NLP tasks other than NER are Part of Speech (POS) tagging and chunking. We present a survey of the latest research from the NER field. But a lot of the mentioned research pertains to sequence labelling tasks other than NER as well. Hence we consider the mentioned research as a comprehensive survey of sequence tagging with a focus on NER. Also, we show research 2006 onwards wherever possible since there has been a shift from linear or log-linear methods (Nadeau and Sekine, 2007) to non-linear methods in the past decade. To the best of our knowledge, NER pre-processing, training and evaluation methods have not been surveyed extensively 2006 onwards.

The first section of the survey briefly covers the evolution of the deep learning framework from linear techniques ie. the advantages and drawbacks of each with respect to training, and performance in different domains, languages and applications. We cover around 150 well-cited English language papers from major conferences and journals. We do not claim this review to be exhaustive or representative of all the research in all languages, but we believe it gives a good estimate of the major research findings in the field.

Sections 2, 3 and 4 detail the computational techniques that have produced state-of-the-art results at various stages of the neural network model in order. In particular, Sections 2 covers feature engineering at the input i.e., syntactical, morphological and contextual fea-

tures. Moreover, Section 2 shows the impact of different distributed representation models of the text on the NER task. There is also a discussion of the structured and unstructured data used and additional preprocessing steps that have shown state-of-the-art NER results. Section 3 discusses the model architecture used in NER including convolutional, recurrent and recursive neural networks. Section 3 also shows the optimisation method, output tag structure and evaluation methods and their impact on the performance of particular techniques. Section 4 summarizes the performance of the discussed techniques on standard datasets used to benchmark NER performance.

2. A Brief History

Researchers have produced a diverse set of techniques to solve the NER task over close to 30 years. (Rau, 1991) solved part of the NER task as we know it at present. However there is previous research in Information Extraction that includes Named Entity Recognition in a more constrained manner than its present form (Besemer and Jacobs, 1987; DeJong et al., 1979; Dyer and Zernik, 1986; Grishman and Kittredge, 2014; Hobbs, 1986; Lytinen and Gershman, 1986; **?**; Young and Hayes, 1985). The initial NER research comprised of hand-crafted rules-based linear models that were overfitted to a very specific structured text corpus eg. military message sets, naval operation reports, M&A news (Jacobs and Rau, 1993) etc. The need for standardisation led to MUC-6 (Grishman and Sundheim, 1996), HUB-4 (Chinchor et al., 1998), MUC-7 and MET-2 (Chinchor and Robinson, 1997), IREX '(Sekine and Isahara, 2000), CONLL (Sang and De Meulder, 2003), ACE (Doddington et al., 2004) and HAREM (Santos et al., 2006). The Language Resources and Evaluation Conference (LREC)1 has also been staging workshops and main conference tracks on the topic since 2000. Supervised learning techniques trained on a large annotated corpus produced state-of-the-art results for NER. The prominent supervised learning methods include Hidden Markov Models (HMM) (Bikel et al., 1997), Decision Trees (Sekine et al., 1998), Maximum Entropy Models (ME) (Borthwick et al., 1998), Support Vector Machines (SVM) (Asahara and Matsumoto, 2003), Conditional Random Fields (CRF) (Lafferty et al., 2001; McCallum and Li, 2003). CRF in particular is one of the most effective NER algorithms. The need in NER to train the probability of the output tags using many leading and lagging non-local sequences makes a discriminative model like CRF more suitable compared to generative models like HMM and stochastic grammar. Although ME models relax the strong independence assumptions that the generative models make, they suffer from a weakness called the label bias problem, wherein the model is biased towards states with few outgoing transitions. CRF solves the problem by jointly considering the weights of different features in all the states instead of normalising the transition probabilities at the state level. (Sarawagi and Cohen, 2004) improved on the existing CRF model by formulating the semi-CRF model that assigns labels to subsequences instead of individual entities without any significant additional computational complexity. '(Passos et al., 2014) used their lexicon-infused skip-gram model on public data to learn high-quality phrase vectors to plug into a log-linear CRF system. Joint models of entity analysis (multitask models) have shown better results on individual tasks compared to models optimised solely for NER. (Durrett and Klein, 2014) developed a structured CRF model that improves the performance of NER (and also that of other entity analysis tasks) by training for 3 tasks eg. coreference resolu-

tion (within-document clustering), named entity recognition (coarse semantic typing), and entity linking (matching to Wikipedia en- tities). (Luo et al., 2015) proposed the JERL (Joint Entity Recognition and Linking) model ie. semi-CRF model extended to capture the dependency between NER and entity linking. Supervised learning methods hit a roadblock because there is a limit to the structured text available for learning discriminative features. This led to Semi-supervised learning methods that utilise the exponentially growing volume of unstructured text ie. webpages to derive contextual information in an unsupervised manner from the seed annotated corpus ie. bootstrapping (Nadeau et al., 2006; Brin, 1998; Collins and Singer, 1999; Yangarber et al., 2002; Riloff et al., 1999; Cucchiarelli and Velardi, 2001; Pasca et al., 2006). (Suzuki et al., 2011) proposed an unsupervised method to create resourceful condensed feature representations from large scale unlabelled data to efficiently train supervised NER systems while still maintaining state-of-the-art results derived from existing higher dimensional semi-supervised solutions.

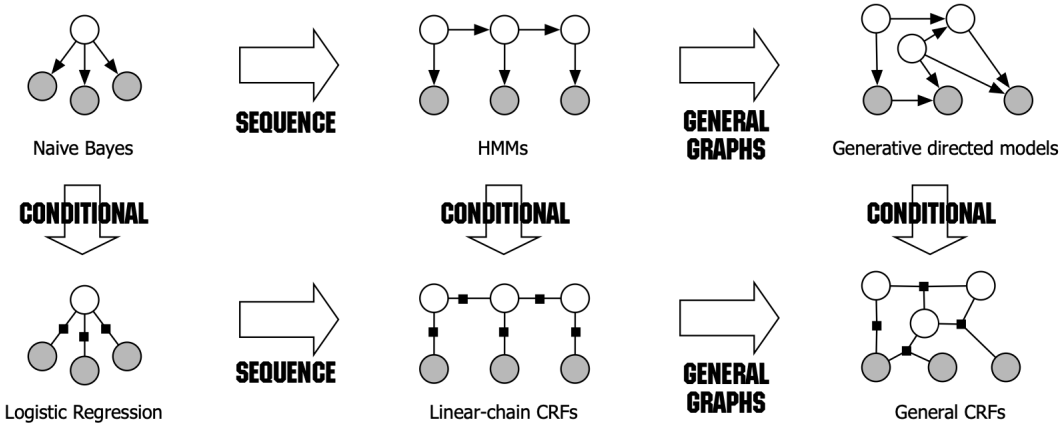

Figure 1: Diagram of the relationship between naive Bayes, logistic regression, HMMs, linear- chain CRFs, generative models, and general CRFs (Sutton and McCallum, 2006)

Unsupervised learning approach emerged mainly as the means to generate additional features from the context of input words to use in conjunction with other NER methods (Evans and Street, 2003; Cimiano and Völker, 2005; Shinyama and Sekine, 2004; Etzioni et al., 2005). More recently, (Lin and Wu, 2009) got state-of-the-art results without using a gazetteer by instead performing k-means clustering over a private database of search engine query logs to use the cluster features to train their CRF model. An exception to the industry application of unsupervised learning technique is the clustering method using lexical resources eg. Wordnet (Alfonseca and Manandhar, 2002) in order assign named entity type from Wordnet itself (location¿country, act¿person etc.). There have been many papers on employing neural networks to do NER in spite of the widespread adoption of CRF for the purpose. (Ratinov and Roth, 2009) used non-local features, a gazetteer ex-

tracted from Wikipedia, and Brown-cluster-like word representations in a perceptron model (Freund and Schapire, 1999). Since multilayer feedforward neural networks are universal approximators, such a neural network could also potentially solve the NER task. (Petasis et al., 2000) used a feed-forward neural network with one hidden layer on NER and achieved state-of-the-art results on the MUC6 dataset. They used gazetteer and POS tags as input features. (Mikolov et al., 2013a) introduced the practise of using the skip-gram model or the continuous bag of words (CBOW) model to create a well-behaving and concise vector representation of words ie. word embeddings, upstream to the neural network model. (Collobert and Weston, 2008) demonstrated the use of pre-trained word embeddings to train a deep neural network to achieve state-of-the-art results in multiple NLP tasks including NER without any additional syntactic features. We discuss word embeddings in further detail in section 3. (Collobert et al., 2011) presented SENNA, which employs a deep FFNN and word embeddings to achieve near state of the art results on NER among other NLP sequence labelling tasks. (Santos et al., 2006) presented their CharWNN network, which extends the neural network of (Collobert et al., 2011) with character level CNNs, and they reported improved performance on Spanish and Portuguese NER. (Hammerton, 2003) attempted NER with a single- direction LSTM network and a combination of word vectors trained using self-organizing maps for lexical representation and principal component analysis for the orthogonal representation of POS and chunk tags. (Huang et al., 2015) used a BLSTM with extensive feature engineering instead of character embeddings for the POS-tagging, chunking, and NER tasks. (Chiu and Nichols, 2015) used a LSTM-CNN model to detect character level and word level features without additional feature engineering for NER. Their model is similar to the RNN-CNNs model introduced by (Labeau et al., 2015) for German POS tagging but uses the advanced LSTM component compared to the RNN. (Lample et al., 2016) showed a BLSTM-CRF model and a stack LSTM model (s-LSTM) (Dyer et al., 2015) to perform the NER task by modeling the output tag dependency using a simple CRF model or a transition model to create and label chunks of input text. Like (Lample et al., 2016), (Yang et al., 2016) also used character level embeddings similar to (Ling et al., 2015). (Yang et al., 2016) also used the deep hierarchical RNN (Cho et al., 2014) for sequence tagging.

3. Features and Data

In this section we look at the input to the NER model. The primary input is the training data. In order to benchmark NER performance, researchers run their model on the CoNLL-2002 and CoNLL- 2003 datasets (Sang, 2002; Sang and De Meulder, 2003) that contain independent named entity labels for English, Spanish, German and Dutch. All datasets contain four different types of named entities: locations, persons, organizations, and miscellaneous entities that do not belong in any of the three previous categories. The miscellaneous category is very diverse as it includes adjectives, like Indian, as well as events, like 1000 Lakes Rally. In the CoNLL-2003 dataset, Named entity tagging of English and German training, development, and test data, was done by hand at the University of Antwerp. Almost all the research validates the hypothesis that NER systems perform better when there is external data. The external knowledge that NER systems use are gazeteers and unlabeled text.

3.1 Unlabelled Text

Recent successful semi-supervised systems (Ando and Zhang, 2005; Suzuki and Isozaki, 2008) have illustrated that unlabeled text can be used to improve the performance of NER systems. (Ratinov and Roth, 2009) uses the implementation of the word class models (Brown et al., 1992) in (Liang, 2005) to obtain word clusters from the Reuters 1996 dataset, a superset of the CoNLL03 NER dataset. (Ratinov and Roth, 2009) produced unique bit strings for each word based on its path from the root in the binary tree that the word class algorithm produces. They multiplied the input features by 4 by path representations of length 4,6,10 and 20 for every word.

(Qi et al., 2009) proposed an iterative Word Class Distribution Learning framework and applied it to a sampled set of Wikipedia webpages. In contrast to self-training (eg. bootstrapping) or co-training methods, the WCDL does not add self-assigned labels that might be subject to learning bias if the model introduces incorrectly labelled examples to the corpus. The WCDL iteratively re-trains a base classifier to build a class label distribution for each word by normalising the predicted labels of the unlabelled corpus. The word class distribution becomes a feature of the base classifier (semi-supervised or supervised NER system) instead of adding many self-annotations, thus making the WCDL highly scalable.

3.2 Gazeteers

An important question in NER research is to address the coverage and the disambiguation of input words using a gazetteer ie. lists of words. There is ample evidence of the improvement of NER performance with the use of high-quality and high-coverage gazetteers (Cohen and Sarawagi, 2004; Torisawa et al., 2007; Toral and Munoz, 2006; Florian et al., 2003). Wikipedia is a great source to construct gazetteers for NER for several reasons. (1) It is manually updated regularly hence there is less chance of missing new information. (2) It maps several variations of spelling or meaning to a relevant entry. For example, 'Manchester' not only refers to the English city of Manchester but also to the football club by the similar name. (3) Wikipedia entries are manually mapped to categories. For example, the entry about the city of Manchester is tagged as a city while Manchester City F.C. is tagged as a football association.

Table 4 summarizes the results of the techniques for injecting external knowledge. Although the additional features were from the superset of the CoNLL03 dataset and the gazetteers were from Wikipedia, the extra features proved to be useful on all datasets. To make the clusters more relevant to this domain, we adopted the following strategy: (1) Construct the feature vectors for 20 million phrases using the web data (2) Run K-Means

clustering on the phrases that appeared in the CoNLL training data to obtain K centroids (3) Assign each of the 20 million phrases to the nearest centroid in the previous step.

$$
\begin{aligned}
[y_s], [y_{s-1:s}], [y_s, w_u]_{u=s-1}^{s+1}, [y_{s-1:s}, w_u]_{u=s-1}^{s+1} \\
[y_s, sfx3_u]_{u=s-1}^{s+1}, [y_{s-1:s}, sfx3_u]_{u=s-1}^{s+1} \\
\{[y_s, wtp_u^t]_{u=s-1}^{s+1}\}_{t=1}^{4}, \{[y_{s-1:s}, wtp_u^t]_{u=s-1}^{s+1}\}_{t=1}^{4} \\
[y_s, w_{u-1:u}]_{u=s}^{s+1}, [y_{s-1:s}, w_{u-1:u}]_{u=s}^{s+1} \\
\{[y_s, wtp_{u-1:u}^t]_{u=s}^{s+1}\}_{t=1}^{3}, \{[y_{s-1:s}, wtp_{u-1:u}^t]_{u=s}^{s+1}\}_{t=1}^{3}
\end{aligned}
\tag{1}
$$

Here, s denotes a position in the input sequence; y_s is a label that indicates whether the token at position s is a named entity as well as its type; w_u is the word at position u; sfx3 is a word's three-letter suffix. $wtp^{t4}_{t=1}$ are indicators of different word types; t=1 denotes punctuation, 2 denotes whether a word is upper cap, lower cap or all caps, 3 denotes a number, 4 denotes that the word has hyphen with different caps before and after the token. Including word unigram features (to capture abbreviated or partial form of complete entities), (Lin and Wu, 2009) had 48 features.

3.3 Word Embeddings

Using pre-trained word embeddings has become a standard feature of NLP tasks including NER. (Collobert et al., 2011) proposed a neural network architecture to construct word-embeddings that forms the primary method to get vector representation of words for training deep learning NLP models for NER. Word embeddings was pioneered by (Mikolov et al., 2013a) who introduced the continuous bag-of-words and the skip-gram models to build highly granular vector representation of words. Glove by (Pennington et al., 2014) is another famous word embedding method which is based on word co-occurences. The frequency matrix is factorised into a lower dimension by minimizing the reconstruction loss after normalisation and smoothing the matrix. The (Mikolov et al., 2013a) method to create word embeddings became widely adopted because this vector representation showed compositionality. Compositionality corresponds to the property of linear semantic inference eg. 'Paris'-'France' + 'Italy'='Rome'.

The CBOW and the continuous skip-gram are both log-linear language models but they differ in a fundamental manner. The CBOW predicts the target word given its context. However the continuous skip gram model predicts words before and after a target word within a given window. The window of neighbouring words that is used as context for the vector representation is a hyperparameter that needs to be optimised. Increasing the window increases the accuracy of the language model but also increases the computational complexity of factoring in distant words in the window. One of the main advantages of the new log-linear models proposed by (Mikolov et al., 2013a) is that they eliminate the hidden layer in feed-forward neural net language models thus reducing computational bottleneck of the language model ie. an optimized single-machine implementation can train on more than 100 billion words in one day. (Mikolov et al., 2013b) further extended the original Skip-gram model for faster training and better quality word representation. They introduced a simple sub-sampling method to speed up the training process as well as ensure more accurate

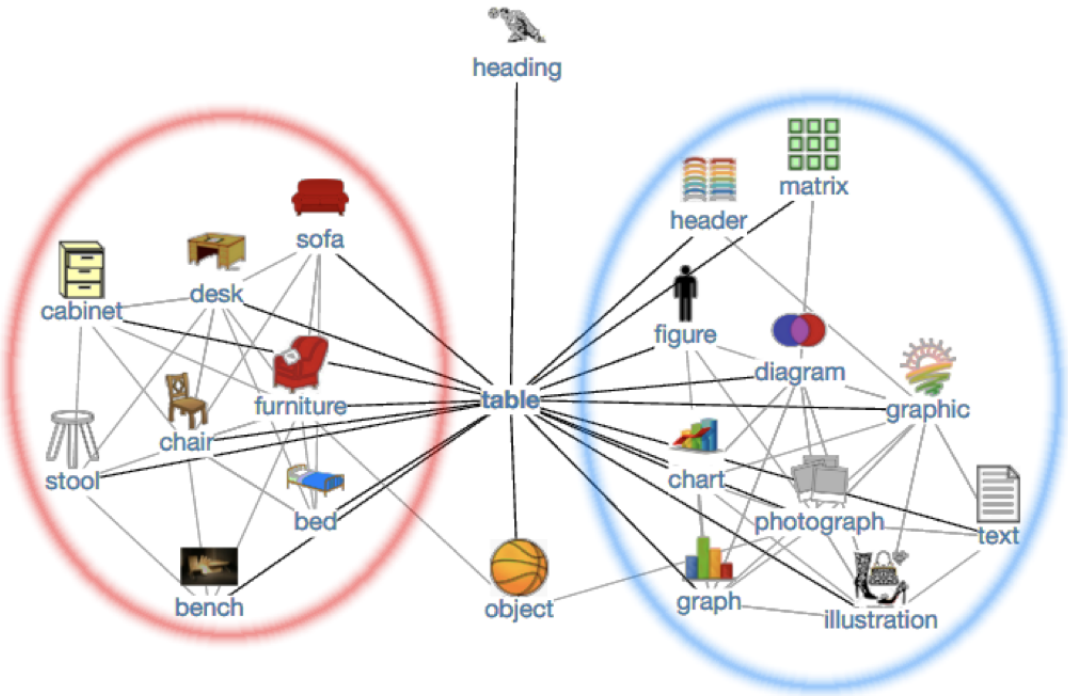

Figure 2: Diagram of the graph obtained from the vector representation of furniture and data sense clusters related to the vector representation of the word 'table' (eg. ego network of 'table') (Pelevina et al., 2017)

representation of less frequent words. The sub-sampling scheme is to discard each word w_i in the training set with probability computed by the formula $P(w_i) = 1 - sqrt(/fract f(w_i))$

Where $f(w_i)$ is the frequency of the word w_i and t is the chosen frequency threshold (typically 10^-5) above which words are sub-sampled significantly. Moreover, they replaced the computationally inefficient softmax method in the output layer with 2 alternatives :1) Hierarchical Softmax (approximation of the full softmax) and 2) Noise Contrastive Estimation (NCE). Negative Sampling or NCE was introduced by (Gutmann and Hyvärinen, 2012) and applied to language modeling by (Mnih and Teh, 2012).

There were caveats to extending the word embeddings to phrase embeddings ie. the football club 'Manchester City' is not the same in meaning as the combination of the words 'Manchester' and 'City'. Mikolov treated phrases as individual tokens in training the log-linear language model. Other methods (Johnson and Zhang, 2015) have obtained the n-gram representation from unlabelled data. Another issue with fixed word embeddings is that it does not account for polysemy ie a single word can be represented by 2 different vectors in 2 different contexts. Traditional word embedding methods such as Word2Vec and Glove consider all the sentences where a word is present in order to create a global

vector representation of that word. However, a word can have completely different senses or meanings in the contexts. For example, let's consider these two sentences - 1) "The enjoyed reading the novel over the weekend" 2) "Scientists have discovered a novel method to treat cancer". The word senses of 'novel 'are different in these two sentences depending on its context. Traditional word embedding methods like word2vec and glove provide the same representation of 'novel' in both the contexts. (Upadhyay et al., 2017) used multilingual data to add another dimension to multi-sense word embeddings. For example, the English word bank, when translated to French provides two different words: banc and banque representing financial and geographical meanings, respectively.

3.4 Character Embeddings

Character-level embeddings find usage in NER to capture the morphological features across languages. Better results on morphologically rich languages are reported in certain NLP tasks. (Santos and Guimaraes, 2015) applied character-level representations, along with word embeddings for NER, achieving state-of-the-art results in Portuguese and Spanish corpora. (Kim et al., 2016) showed positive results on building a neural language model using only character embeddings. (Ma et al., 2016) exploited several embeddings, including character trigrams, to incorporate prototypical and hierarchical information for learning pre-trained label embeddings in the context of NER. Chinese is another morphologically rich language in which character embeddings have shown better performance compared to word embeddings for deep learning sequence labelling models (Zheng et al., 2013).

Word embeddings do not convey the semantic meaning and other internal information that character embeddings provide. Hence character embeddings are able to deduce the meaning of unknown words by mapping the meaning of such words to that of the compositional characters or sub-words. Therefore character embeddings solve the problem of identifying out-of-vocabulary (OOV) words eg. words that are not present in the input corpus for tasks like part-of-speech tagging and language modeling (Ling et al., 2015) or dependency parsing (Ballesteros et al., 2015). Character embeddings provide a feasible method to represent word types. Character embeddings have found widespread use as they avoid the additional dimensions that are introduced to solve the OOV problem by strictly using word representations. Chen et al. (2015) shows that introducing character embeddings to word embeddings in the Chinese language results in more informed word representation eg. better results in word relatedness and analogical reasoning tasks. Character embeddings also have the advantage of becoming relevant to the task, language and domain at hand (Vosoughi et al., 2016; Saxe and Berlin, 2017). Bojanowski et al. (2017) tried to improve the popular skip-gram method by representing words as bag-of-character n-grams. Therefore their work solved the limitations of word embeddings with the efficient skip-gram model. Their fastText library (Joulin et al., 2016, 2017) produces text classification models within 100kB without compromising on accuracy or speed. The typical architecture involved in deriving word embeddings from characters initialises a character lookup table with random representation of each character. The character embeddings of every character in the sequence then passes through forward and backward LSTMs. The character representations from the biLSTM then concatenate with the word representation from a word lookup table. The forward LSTM represents the suffix of the word while the backward LSTM represents

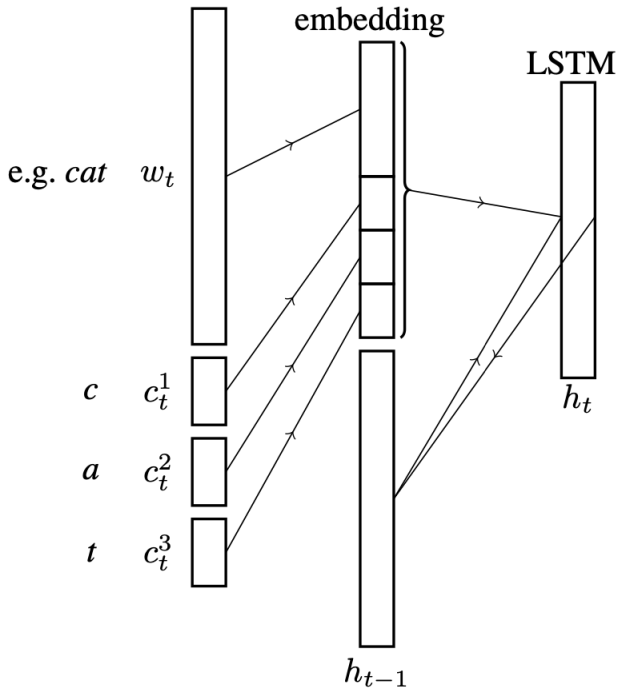

Figure 3: Merging of the word and character embeddings of the word 'cat' in a LSTM LM (eg. language model) (Verwimp et al., 2017)

the prefix of the word. LSTM networks model long-range dependencies in temporal sequences better than traditional RNNs (Sak et al., 2014). However, LSTM networks do not capture the semantic difference between the prefix and the suffix of a word that biLSTMs do. Convolutional neural networks have also been proposed to discover position invariant features of characters from words (**?**). Efficient word selection further tackles the issue of non-compositional words or unknown words or ambiguous characters.

Apart from character embeddings, different approaches have been proposed for OOV handling. (Herbelot and Baroni, 2017) provided an essential component of incremental concept learning by initializing the unknown words as the greedy combination of the context words and refining these words with a high learning rate. However, their approach is yet to be tested on typical NLP tasks where the processing of context requires modulation. Pinter et al. (2017) provided a character-based model that does not require re-training of character embeddings from the original corpus thus saving processing time. This allowed them to learn a compositional mapping form character to word embedding, thus efficiently tackling the OOV problem.

Despite the ever growing popularity of distributional vectors, there are also limitations of these vectors. For example, (Lucy and Gauthier, 2017) has recently tried to evaluate how

well the word vectors predict the perceptual and conceptual features of distinct concepts, using a semantic norms dataset that has been created by human participants as a benchmark. The authors have discovered severe limitations of distributional models in making the fundamental understanding of the concepts behind the words. A possible direction for mitigating these deficiencies will be grounded learning (Niekum et al., 2013).

3.5 Model Specification

3.5.1 Convolutional Neural Networks

CNNs found use in extracting feature vector representations from words (Collobert et al., 2011) using a lookup table learned by backpropagation. CNNs therefore seemed to be the natural choice in extracting higher order features from individual words in an input sequence of variable length. There are 2 approaches : (1) window approach and (2) (convolutional) sentence approach. The window approach assumes that the tag assigned to an individual word depends on its context (ie. words occurring before and after the given word). The window approach is more suitable for sequence tagging tasks like NER.

$$f_\theta^l = \langle LT_W([w]_l^T)\rangle_t^{d_{win}} = \begin{pmatrix} \langle W\rangle^l_{[w]_{t-d_{win}/2}} \\ . \\ . \\ . \\ \langle W\rangle^l_{[w]_t} \\ . \\ . \\ . \\ \langle W\rangle^l_{[w]_{t+d_{win}/2}} \end{pmatrix} \tag{2}$$

$$f_\theta^l = W^l f_\theta^{l-1} + b^l \tag{3}$$

Here $W^l \in R^{n_h^l \times n_h^{l-1}}$ and $b^l \in R^{n_h^l}$ are trained using backpropagation. Here n_h^l is a hyper-parameter that indicates the number of hidden units in the l^{th} layer. The fixed size vector input can be passed through multiple linear transformation layers as shown above. In order to capture the higher-order features from the input sequence, there are layers of ''hard" hyperbolic tangent functions. The "hard" hyperbolic tangent is computationally efficient compared to the exact hyperbolic tangent function while preventing overfitting. One caveat of the window feature is that the context of the words at the beginning and the end of a sentence are not well-defined. Hence there are padding words half the size of the window at both the beginning and the end of input sentences akin to the start and stop indicators in sequence models.

$$[f_\theta^l]_i = HardTanh([f_\theta^{l-1}]_i) \tag{4}$$

$$Hardtanh(x) = \begin{cases} -1 & if\ x < -1 \\ x & if\ -1 \leq x \leq 1 \\ 1 & if\ x > -1 \end{cases} \tag{5}$$

The use of CNNs for sentence modeling traces back to (Collobert and Weston, 2008). This work used multi-task learning to output multiple predictions for NLP tasks such as POS tags, chunks, named-entity tags, semantic roles, semantically-similar words and a language model. A look-up table was used to transform each word into a vector of user-defined dimensions. Thus, an input sequence $/s_1\ , s_2\ , ...s_n\ /$ of n words was transformed into a series of vectors $/ws_1\ , ws_2\ , ...ws_n\ /$ by applying the look-up table to each of its words.

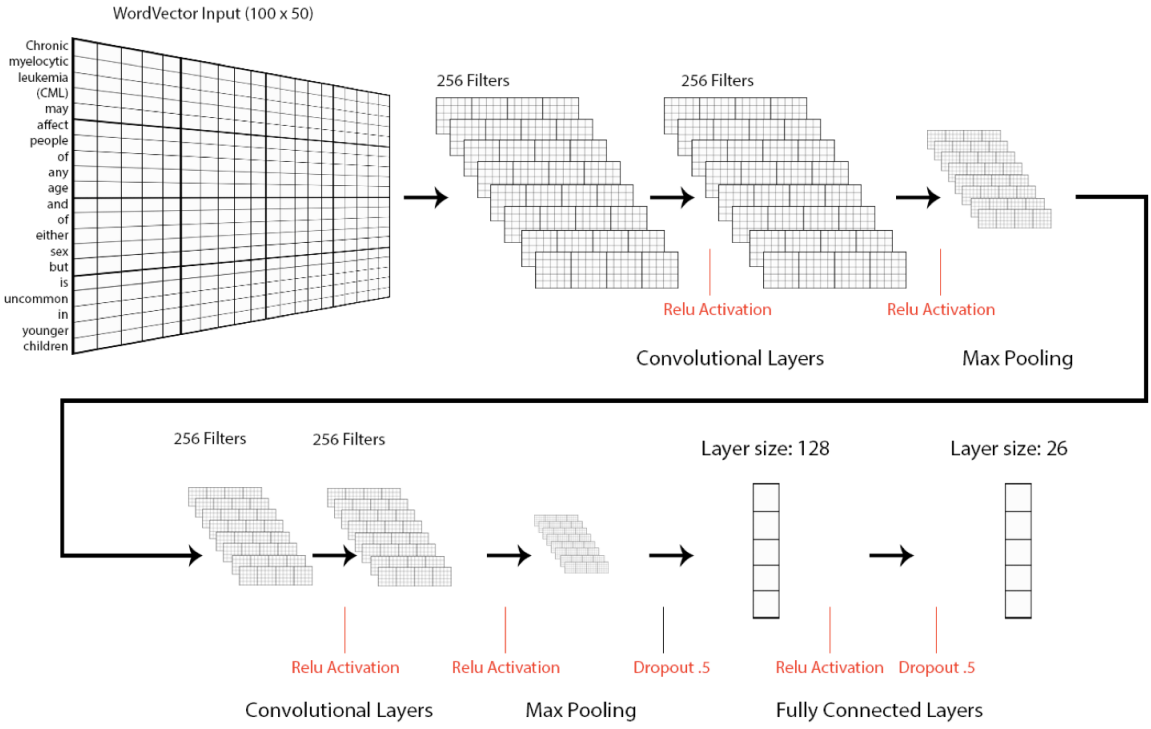

Figure 4: Example of a CNN architecture for medical text classification (Hughes et al., 2017)

This can be thought of as a primitive word embedding method whose weights were learned in the training of the network. In (Collobert et al., 2011), Collobert et al extended their work to propose a general CNN-based framework to solve a plethora of NLP tasks. Both these works triggered a huge popularization of CNNs amongst NLP researchers. Given that CNNs had already shown their mettle for computer vision tasks, it was easier for people to believe in their performance.

CNNs have the ability to extract salient n-gram features from the input sentence to create an informative latent semantic representation of the sentence for downstream tasks. This application was pioneered by (Collobert et al., 2011; Kalchbrenner et al., 2014; Kim, 2014) which led to a huge proliferation of CNN-based networks in the succeeding literature. Below, we describe the working of a simple CNN-based sentence modeling network:

3.5.2 Recurrent Neural Network

Recurrent Neural Network (Elman, 1990) is another form of deep neural network architecture. In general, CNNs are used to represent position-invariant functions (eg bag of words) while RNNs represent a sequential architecture (eg. sentences, paragraphs etc.). It seems obvious that RNNs would be more suitable for sequence modelling tasks like NER compared to the hierarchical CNNs. While we saw that the window approach helps to tackle sequence inputs in CNN, there is no way to model dependencies on context beyond that which is fixed in cross-validation. RNNs help to capture dependencies on words or sentences beyond the fixed range of CNNs. The simple RNN has no gating mechanism. RNN is a network that is unfolded across time thus providing a spatial representation of memory. For a given input, the RNN calculates a hidden state as follows:

$$[s^t_{\theta,l}]_i = g_{\theta,l}([f^t_{\theta,l}]_i) \tag{6}$$

$$[f^t_{\theta,l}]_i = W^l[x^t_l]_i + [s^{t-1}_{\theta,l}]_i \tag{7}$$

Here $[s^t_{\theta,l}]_i$ is the hidden state at time t and layer l for the i_th unit of the input sequence. $g_{\theta,l}$ is a nonlinear function (eg. tanh etc.) at the layer which takes as input $f^t_{\theta,l}$. $[x^t_l]_i$ is the ith unit (word, sentence etc.) of the input sequence at time t and $W^l \in R^{d_{s^t_{\theta,l}} \times d_{x^t_l}}$ are the weights that are learned through training. The hidden state in a simple RNN can be considered its memory component. However, simple RNNs suffer from the vanishing gradient problem that makes it difficult to learn the weights in the previous time steps using backpropagation. Therefore simple RNNs are augmented with a gating mechanism to overcome the convergence problem. RNN variants with gating mechanism that are most popular in NER are long-short term memory (LSTM) and Gated Recurrent Units (GRUs).

3.5.3 Long Short Term Memory

The novelty of LSTM (Hochreiter and Schmidhuber, 1997) is that it is able to bridge long time intervals (to quickly learn the slowly changing weights many time steps back eg. long term memory) as well as preserve the recent inputs (eg. short term memory). Moreover, the LSTM architecture ensures constant re-weighting thus avoiding the explosion or vanishing of error flow through the hidden states.

$$i_t = \sigma(x_t U^i + h_{t-1} W^i + b_i) \tag{8}$$

$$f_t = \sigma(x_t U^f + h_{t-1} W^f + b_f) \tag{9}$$

$$o_t = \sigma(x_t U^o + h_{t-1} W^o + b_o) \tag{10}$$

$$q_t = tanh(x_t U^q + h_{t-1} W^q + b_q) \tag{11}$$

$$p_t = f_t \times p_{t-1} + i_t \times q_t \tag{12}$$

$$h_t = o_t \times tanh(p_t) \tag{13}$$

LSTM has three gates: input gate i_t, forget gate f_t and output gate o_t that are outputs of the sigmoid function over the input x_t and the preceding hidden state h_{t-1}. In order to generate the hidden state at current step t, it first generates a temporary variable q_t by running the non-linear tanh function on the input x_t and the previous hidden state h_{t-1}. The LSTM then calculates an updated history variable at time t p_t as a linear combination of the previous history state p_{t-1} and the current temporary variable q_t scaled by the current forget gate f_t and the current input gate i_t respectively. Finally LSTM runs the tanh on p_t and scales it by the current output gate o_t to get the updated hidden state h_t .

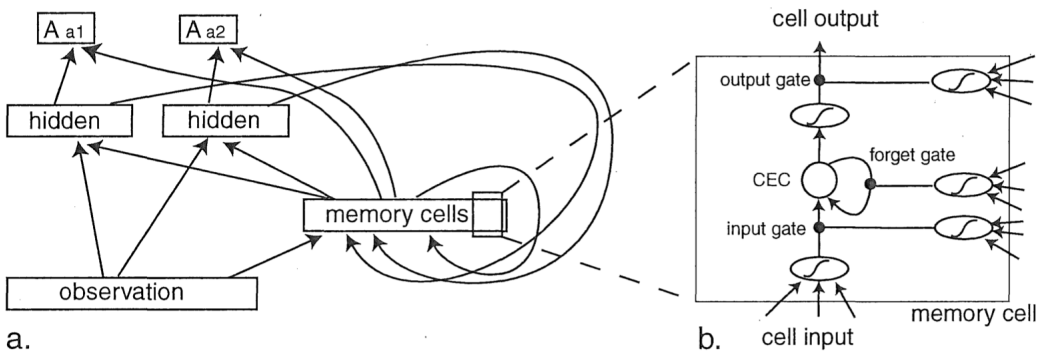

Figure 5: a. General LSTM architecture b. One memory cell (Bakker, 2002)

While LSTM is good at approximating the dependency of the current unit of the sequence with previous units, it does not take into account the dependency of the current unit on units to its right in the sequence. (Lample et al., 2016) solved this problem by implementing a bi-directional LSTM (Graves and Schmidhuber, 2005). In other words, there are 2 independent LSTMs that are trained separately using the segment of the input sequence to the left and the right of the target word. The model gets the full representation of the target word by concatenating the left and right context representations of the forward and the backward LSTM respectively ie the hidden state $h_t = [\overrightarrow{h}_t; \overleftarrow{h}_t]$.

3.5.4 Gated Recurrent Unit

Gated Recurrent Unit (Cho et al., 2014) is a more recent and less complex variant of the RNN compared to the LSTM.

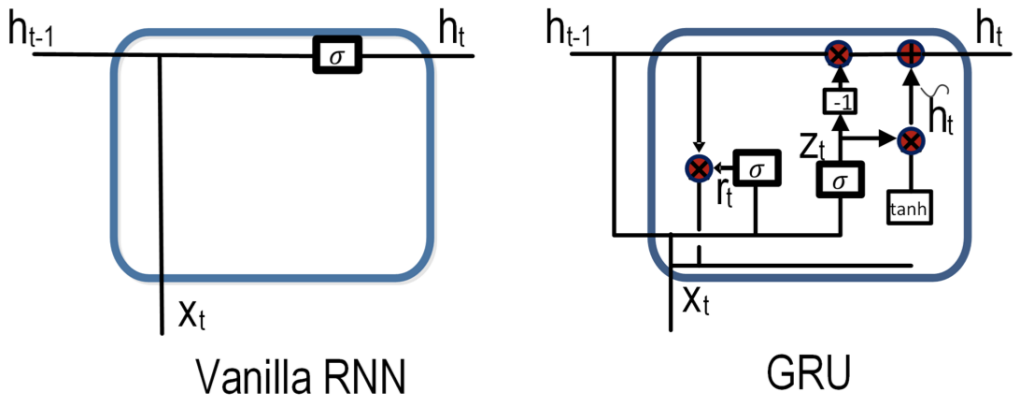

Figure 6: Vanilla RNN and GRU architecture (single unit shown for simplicity) with a reset gate r (adjusts the incorporation of new input with the previous memory), an update gate z (controls the preservation of the previous memory), current hidden output h_t and previous hidden output h_{t-1} (Zhao et al., 2017)

The GRU is similar to the LSTM as it modulates the error flow thus avoiding vanishing gradients (Bengio et al., 1994). However, GRU has many crucial differences with the LSTM. GRUs do not have separate memory cells like LSTMs. Therefore GRUs lack the controlled exposure to the memory content (ie the output gate of LSTMs). Unlike LSTMs, GRUs expose the full content without any control. Moreover, GRUs control the information flow from the previous activation to the current, candidate activation. On the other hand, LSTMs compute the latest memory content without any control over the historical information flow. The GRU works using the following equations:

$$h_t^j = (1 - z_t^j)h_{t-1}^j + z_t^j \bar{h}_t^j \tag{14}$$

$$z_t^j = \sigma(W_z x_t + U_z h_{t-1}) \tag{15}$$

$$\bar{h}_t^j = tanh(W x_t + U(r_t \odot h_{t-1})) \tag{16}$$

$$r_t^j = \sigma(W_r x_t + U_r h_{t-1}) \tag{17}$$

h_t^j is the activation of the GRU at level j at time t. $\bar{h}_t^j$ is the candidate activation of the GRU (Bahdanau et al., 2014) at level j at time t. The update gate z_t^j decides how much the unit updates its activation. The reset gate r_t^j is computed similar to the update gate.

3.5.5 Results

Tables 1 and 2 show the F1 scores of state of the art neural network models in chronological order for NER on the dev as well as test data set from CoNLL-2003 shared task and the

OntoNotes 5.0 NER task (English). Tables 3 4 show the F1 scores of the state of the art models for the CoNLL 2002 Spanish and Dutch NER shared tasks.

Model	**CoNLL-2003**			**OntoNotes 5.0**		
	Prec.	**Recall**	**F1**	**Prec.**	**Recall**	**F1**
Collobert et al. (2011)	-	-	88.67	-	-	-
Collobert et al. (2011)+lexicon	-	-	89.59	-	-	-
Huang et al. (2015)	-	-	84.26	-	-	-
Huang et al. (2015)+Senna + gazetteer	-	-	90.10	-	-	-
Chiu and Nichols (2015)	83.48	83.28	83.28(± 0.20)	82.58	82.49	82.53(± 0.40)
Chiu and Nichols (2015) + emb	90.75	91.08	90.91(± 0.20)	85.99	86.36	86.17(± 0.22)
Chiu and Nichols (2015) + emb + lex	91.39	91.85	91.62(± 0.33)	86.04	86.53	86.28(± 0.26)
Ma and Hovy (2016)	-	-	91.37	-	-	-
Lample et al. (2016) (Bi-LSTM)	-	-	89.15	-	-	-
Lample et al. (2016) (Bi-LSTM-CRF)	-	-	90.94	-	-	-
Hu et al. (2016)	-	-	91.18	-	-	-

Table 1: Table to test captions and labels

There are 22 model results in Tables 1 and 2 while there are 8 model results in Tables 3 and 4. For the purpose of presentation, the neural models are divided into 2 tables with 11 results in each table. Tables 3 and 4 show the F1 scores of the linear and the log-linear models (ie. does not involve neural networks) for NER. The latest NER model that does not have any form of neural network in its architecture was produced in 2015. All the NER models that have produced state of the art results since 2015 use neural network architecture. Therefore, it is unlikely that linear methods will be able to compete with the results produced by deep learning models for NER and other sequence tagging tasks.

It is also clear from Table 1 that the use of character-level and word-level knowledge (eg. gazetteers, benchmark corpus like (Chelba et al., 2013), LM ie language model) benefits neural models. Specifically word embeddings (Mikolov et al., 2013a; Pennington

Model	CoNLL-2003			OntoNotes 5.0		
	Prec.	**Recall**	**F1**	**Prec.**	**Recall**	**F1**
Yang et al. (2016) - no word embeddings	-	-	77.20	-	-	-
Yang et al. (2016) - no char GRU	-	-	88.00	-	-	-
Yang et al. (2016) - no gazetteer	-	-	90.96	-	-	-
Yang et al. (2016)	-	-	91.20	-	-	-
Rei (2017)	-	-	87.38(±0.36)	-	-	-
Strubell et al. (2017)	-	-	90.54	-	-	-
Yang et al. (2017)	-	-	91.20	-	-	-
Yang et al. (2017) + CoNLL 2000/PTB-POS	-	-	91.26	-	-	-
Peters et al. (2017)(word embedding)	-	-	90.87(±0.13)	-	-	-
Peters et al. (2017)(LM embedding)	-	-	90.79(±0.15)	-	-	-
Peters et al. (2017) + 1B Word dataset	-	-	91.62(±0.23)	-	-	-
Peters et al. (2017) + 1B Word dataset + 4096-8192-1024	-	-	91.93(± 0.19)	-	-	-

Table 2: Table to test captions and labels

et al., 2014) and sub-word or character embeddings are used as additional features to the input text corpus for co-training.

Moreover, most of the recent LSTM models (eg. LSTM-CRF (Huang et al., 2015; Lample et al., 2016), LSTM-CNNs (Chiu and Nichols, 2015)) utilize syntax (eg. character type, capitalization), lexicon, context, POS features as well as preprocessing to suit the NER task. **?** achieved a high F! score (91.2) by using hand-crafted features like word, character, POS, chunk and stemmed n-grams, Brown and WordNet clusters as well as dictionaries from external knowledge bases like Wikipedia and Freebase.

Such additional information also finds use as pre-trained embeddings to downstream models. Although the pre-trained models are not task-specific, they improve the performance of NER models. For example, **?** reports a significant improvement (Wilcoxon

Model	CoNLL-2003			OntoNotes 5.0		
	Prec.	**Recall**	**F1**	**Prec.**	**Recall**	**F1**
Chieu and Ng (2002)	-	-	88.31	-	-	-
Florian et al. (2003)	-	-	88.76	-	-	-
Ando and Zhang (2005)(dev)	-	-	93.15	-	-	-
Ando and Zhang (2005)(test)	-	-	89.31	-	-	-
Finkel et al. (2005)	95.10	78.30	85.90	-	-	-
Suzuki and Isozaki (2008)(dev)	-	-	94.48	-	-	-
Suzuki and Isozaki (2008)(test)	-	-	89.92	-	-	-

Table 3: Table to test captions and labels

Model	CoNLL-2003			OntoNotes 5.0		
	Prec.	**Recall**	**F1**	**Prec.**	**Recall**	**F1**
Ratinov and Roth (2009)	91.20	90.50	90.80	82.00	84.95	83.45
Lin and Wu (2009)	-	-	90.90	-	-	-
Finkel and Manning (2009)	-	-	-	84.04	80.86	82.42
Suzuki et al. (2011)	-	-	91.02	-	-	-
Sil and Yates (2013)	86.80	89.50	88.20	-	-	-
Passos et al. (2014)	-	-	90.90	-	-	82.24
Durrett and Klein (2014)	-	-	-	85.22	82.89	84.04
Luo et al. (2015) (NER features)	90.00	89.90	89.90	-	-	-
Luo et al. (2015) (NER + EL features)	91.50	91.40	91.20	-	-	-

Table 4: Table to test captions and labels

ranked sum test, p ¡ 0.001) when they train their model with word embeddings (Collobert et al., 2011) compared to random embeddings regardless of the additional features used for training. Co-training strategies are more favorable compared to pre-training strategies since the former requires a less complex neural network compared to the latter. Therefore co-training is more efficient because it does not require any additional corpus and hence the training time is also less.

Model	**CoNLL 2002 Dutch NER**	**CoNLL 2002 Spanish NER**
	F1 (%)	**F1 (%)**
Carreras et al. (2002)	77.05	81.39
Nothman et al. (2013)	78.60	-
Gillick et al. (2015)	82.84	82.95
Lample et al. (2016)	81.74	85.75
Yang et al. (2016) - no word embeddings	67.36	73.34
Yang et al. (2016) - no char GRU	77.76	83.03
Yang et al. (2016)	85.00	84.69
Yang et al. (2016) + joint training	85.19	85.77

Table 5: Table to test captions and labels

Efforts to increase NER F1 score by pre-training, co-training or joint training hss come at a cost. With respect to a generic neural network architecture, additional training layers significantly increases the complexity and the training time of the NER models making the use of these models difficult in practice. Strubell et al. (2017) introduced the use of dilated convolutions that gave close to state of the art results while improving efficiency over existing models by processing a larger input window at a time to model context in a parallelized manner. Another aspect that is evident from Table 1 is that CRF is very suitable as the output layer of neural network NER models.

Acknowledgments

We would like to acknowledge support for this project from the Dr. Geraldo Filgueiras

References

Enrique Alfonseca and Suresh Manandhar. An unsupervised method for general named entity recognition and automated concept discovery. In *Proceedings of the 1st international conference on general WordNet, Mysore, India*, pages 34–43, 2002.

Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6(Nov):1817–1853, 2005.

Masayuki Asahara and Yuji Matsumoto. Japanese named entity extraction with redundant morphological analysis. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 8–15, 2003.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Bram Bakker. Reinforcement learning with long short-term memory. In *Advances in neural information processing systems*, pages 1475–1482, 2002.

Miguel Ballesteros, Chris Dyer, and Noah A Smith. Improved transition-based parsing by modeling characters instead of words with lstms. *arXiv preprint arXiv:1508.00657*, 2015.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

David J Besemer and Paul S Jacobs. Flush: a flexible lexicon design. In *25th Annual Meeting of the Association for Computational Linguistics*, pages 186–192, 1987.

D Bikel, S Miller, R Schwartz, and R Weischedel. Nymble:, a highperformance learning text finder. In *of the Second Conference on Empirical Methods in Natural Language Processing*, pages 109–116, 1997.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

Andrew Borthwick, John Sterling, Eugene Agichtein, and Ralph Grishman. Exploiting diverse knowledge sources via maximum entropy in named entity recognition. In *Sixth Workshop on Very Large Corpora*, 1998.

Sergey Brin. Extracting patterns and relations from the world wide web. In *International Workshop on The World Wide Web and Databases*, pages 172–183. Springer, 1998.

Peter F Brown, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer. Class-based n-gram models of natural language. *Computational linguistics*, 18 (4):467–480, 1992.

Xavier Carreras, Lluís Màrquez, and Lluís Padró. Named entity extraction using adaboost. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, 2002.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*, 2013.

Xinxiong Chen, Lei Xu, Zhiyuan Liu, Maosong Sun, and Huanbo Luan. Joint learning of character and word embeddings. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

Hai Leong Chieu and Hwee Tou Ng. Named entity recognition: a maximum entropy approach using global information. In *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002.

Nancy Chinchor and Patricia Robinson. Muc-7 named entity task definition. In *Proceedings of the 7th Conference on Message Understanding*, volume 29, pages 1–21, 1997.

Nancy Chinchor, Patty Robinson, and Erica Brown. Hub-4 named entity task definition version 4.8. *Available by ftp from www. nist. gov/speech/hub4_98*, 1998.

Jason PC Chiu and Eric Nichols. Named entity recognition with bidirectional lstm-cnns 2015. *arXiv preprint arXiv:1511.08308*, 2015.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Philipp Cimiano and Johanna Völker. text2onto. In *International conference on application of natural language to information systems*, pages 227–238. Springer, 2005.

William W Cohen and Sunita Sarawagi. Exploiting dictionaries in named entity extraction: combining semi-markov extraction processes and data integration methods. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98, 2004.

Michael Collins and Yoram Singer. Unsupervised models for named entity classification. In *1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1999.

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

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(ARTICLE):2493–2537, 2011.

Alessandro Cucchiarelli and Paola Velardi. Unsupervised named entity recognition using syntactic and semantic contextual evidence. *Computational Linguistics*, 27(1):123–131, 2001.

Gerald Francis DeJong et al. Skimming stories in real time: An experiment in integrated understanding. Technical report, YALE UNIV NEW HAVEN CONN DEPT OF COMPUTER SCIENCE, 1979.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. The automatic content extraction (ace) program-tasks, data, and evaluation. In *Lrec*, volume 2, pages 837–840. Lisbon, 2004.

Greg Durrett and Dan Klein. A joint model for entity analysis: Coreference, typing, and linking. *Transactions of the association for computational linguistics*, 2:477–490, 2014.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A Smith. Transition-based dependency parsing with stack long short-term memory. *arXiv preprint arXiv:1505.08075*, 2015.

Michael G Dyer and Uri Zernik. Encoding and acquiring meanings for figurative phrases. In *24th Annual Meeting of the Association for Computational Linguistics*, pages 106–111, 1986.

Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

Hakan Erdogan. Sequence labeling: Generative and discriminative approaches. ICMLA, 2010.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S Weld, and Alexander Yates. Unsupervised named-entity extraction from the web: An experimental study. *Artificial intelligence*, 165(1):91–134, 2005.

Richard Evans and Stafford Street. A framework for named entity recognition in the open domain. *Recent advances in natural language processing III: selected papers from RANLP*, 260(267-274):110, 2003.

Jenny Rose Finkel and Christopher D Manning. Joint parsing and named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 326–334, 2009.

Jenny Rose Finkel, Trond Grenager, and Christopher D Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 363–370, 2005.

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. Named entity recognition through classifier combination. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, pages 168–171, 2003.

Yoav Freund and Robert E Schapire. Large margin classification using the perceptron algorithm. *Machine learning*, 37(3):277–296, 1999.

Dan Gillick, Cliff Brunk, Oriol Vinyals, and Amarnag Subramanya. Multilingual language processing from bytes. *arXiv preprint arXiv:1512.00103*, 2015.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610, 2005.

Ralph Grishman and Richard Kittredge. *Analyzing language in restricted domains: sublanguage description and processing*. Psychology Press, 2014.

Ralph Grishman and Beth M Sundheim. Message understanding conference-6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, 1996.

Michael U Gutmann and Aapo Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *The journal of machine learning research*, 13(1):307–361, 2012.

James Hammerton. Named entity recognition with long short-term memory. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, pages 172–175, 2003.

Aurélie Herbelot and Marco Baroni. High-risk learning: acquiring new word vectors from tiny data. *arXiv preprint arXiv:1707.06556*, 2017.

Jerry R Hobbs. The finite string newsletter: Site report another from the darpa series, overview of the tacitus project. *Computational Linguistics*, 12(3), 1986.

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

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. Harnessing deep neural networks with logic rules. *arXiv preprint arXiv:1603.06318*, 2016.

Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

Mark Hughes, Irene Li, Spyros Kotoulas, and Toyotaro Suzumura. Medical text classification using convolutional neural networks. *Stud Health Technol Inform*, 235:246–50, 2017.

Paul S Jacobs and Lisa F Rau. Innovations in text interpretation. *Artificial Intelligence*, 63(1-2):143–191, 1993.

Rie Johnson and Tong Zhang. Semi-supervised convolutional neural networks for text categorization via region embedding. In *Advances in neural information processing systems*, pages 919–927, 2015.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. Fasttext.zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*, 2016.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics, April 2017.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander Rush. Character-aware neural language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Matthieu Labeau, Kevin Löser, and Alexandre Allauzen. Non-lexical neural architecture for fine-grained pos tagging. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 232–237, 2015.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.

Percy Liang. *Semi-supervised learning for natural language*. PhD thesis, Massachusetts Institute of Technology, 2005.

Dekang Lin and Xiaoyun Wu. Phrase clustering for discriminative learning. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1030–1038, 2009.

Wang Ling, Tiago Luís, Luís Marujo, Ramón Fernandez Astudillo, Silvio Amir, Chris Dyer, Alan W Black, and Isabel Trancoso. Finding function in form: Compositional character models for open vocabulary word representation. *arXiv preprint arXiv:1508.02096*, 2015.

Li Lucy and Jon Gauthier. Are distributional representations ready for the real world? evaluating word vectors for grounded perceptual meaning. *arXiv preprint arXiv:1705.11168*, 2017.

Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. Joint entity recognition and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 879–888, 2015.

Steven L Lytinen and Anatole Gershman. Atrans automatic processing of money transfer messages. In *AAAI*, volume 86, pages 1089–1093, 1986.

Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*, 2016.

Yukun Ma, Erik Cambria, and Sa Gao. Label embedding for zero-shot fine-grained named entity typing. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 171–180, 2016.

Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. 2003.

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

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

Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. *arXiv preprint arXiv:1206.6426*, 2012.

David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

David Nadeau, Peter D Turney, and Stan Matwin. Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In *Conference of the Canadian society for computational studies of intelligence*, pages 266–277. Springer, 2006.

Scott Niekum, Sachin Chitta, Andrew G Barto, Bhaskara Marthi, and Sarah Osentoski. Incremental semantically grounded learning from demonstration. In *Robotics: Science and Systems*, volume 9, pages 10–15607. Berlin, Germany, 2013.

Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R Curran. Learning multilingual named entity recognition from wikipedia. *Artificial Intelligence*, 194:151–175, 2013.

Marius Pasca, Dekang Lin, Jeffrey Bigham, Andrei Lifchits, and Alpa Jain. Organizing and searching the world wide web of facts-step one: the one-million fact extraction challenge. In *AAAI*, volume 6, pages 1400–1405, 2006.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. Lexicon infused phrase embeddings for named entity resolution. *arXiv preprint arXiv:1404.5367*, 2014.

Maria Pelevina, Nikolay Arefyev, Chris Biemann, and Alexander Panchenko. Making sense of word embeddings. *arXiv preprint arXiv:1708.03390*, 2017.

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

G Petasis, S Petridis, G Paliouras, V Karkaletsis, SJ Perantonis, and CD Spyropoulos. Symbolic and neural learning for named-entity recognition. In *Proceedings of the Symposium on Computational Intelligence and Learning*, pages 58–66. Citeseer, 2000.

Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. Semi-supervised sequence tagging with bidirectional language models. *arXiv preprint arXiv:1705.00108*, 2017.

Yuval Pinter, Robert Guthrie, and Jacob Eisenstein. Mimicking word embeddings using subword rnns. *arXiv preprint arXiv:1707.06961*, 2017.

Yanjun Qi, Ronan Collobert, Pavel Kuksa, Koray Kavukcuoglu, and Jason Weston. Combining labeled and unlabeled data with word-class distribution learning. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1737–1740, 2009.

Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, 2009.

Lisa F Rau. Extracting company names from text. In *Proceedings the Seventh IEEE Conference on Artificial Intelligence Application*, pages 29–30. IEEE Computer Society, 1991.

Marek Rei. Semi-supervised multitask learning for sequence labeling. *arXiv preprint arXiv:1704.07156*, 2017.

Ellen Riloff, Rosie Jones, et al. Learning dictionaries for information extraction by multi-level bootstrapping. In *AAAI/IAAI*, pages 474–479, 1999.

Hasim Sak, Andrew W Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. 2014.

Erik F Sang. Tjong kim (2002)."introduction to the conll-2002 shared task: Language-independent named entity recognition". In *COLING-02: The 6th Conference on Natural Language Learning*, 2002.

Erik F Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*, 2003.

Cicero Nogueira dos Santos and Victor Guimaraes. Boosting named entity recognition with neural character embeddings. *arXiv preprint arXiv:1505.05008*, 2015.

Diana Santos, Nuno Seco, Nuno Cardoso, and Rui Vilela. Harem: An advanced ner evaluation contest for portuguese. In *quot; In Nicoletta Calzolari; Khalid Choukri; Aldo Gangemi; Bente Maegaard; Joseph Mariani; Jan Odjik; Daniel Tapias (ed) Proceedings of the 5 th International Conference on Language Resources and Evaluation (LREC'2006)(Genoa Italy 22-28 May 2006)*, 2006.

Sunita Sarawagi and William W Cohen. Semi-markov conditional random fields for information extraction. *Advances in neural information processing systems*, 17:1185–1192, 2004.

Joshua Saxe and Konstantin Berlin. expose: A character-level convolutional neural network with embeddings for detecting malicious urls, file paths and registry keys. *arXiv preprint arXiv:1702.08568*, 2017.

Satoshi Sekine and Hitoshi Isahara. Irex: Ir & ie evaluation project in japanese. In *LREC*, pages 1977–1980. Citeseer, 2000.

Satoshi Sekine, Ralph Grishman, and Hiroyuki Shinnou. A decision tree method for finding and classifying names in japanese texts. In *Sixth workshop on very large corpora*, 1998.

Yusuke Shinyama and Satoshi Sekine. Named entity discovery using comparable news articles. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 848–853, 2004.

Avirup Sil and Alexander Yates. Re-ranking for joint named-entity recognition and linking. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2369–2374, 2013.

Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. Fast and accurate sequence labeling with iterated dilated convolutions. 2017.

Charles Sutton and Andrew McCallum. An introduction to conditional random fields for relational learning. *Introduction to statistical relational learning*, 2:93–128, 2006.

Jun Suzuki and Hideki Isozaki. Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *Proceedings of ACL-08: HLT*, pages 665–673, 2008.

Jun Suzuki, Hideki Isozaki, and Masaaki Nagata. Learning condensed feature representations from large unsupervised data sets for supervised learning. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 636–641, 2011.

Antonio Toral and Rafael Munoz. A proposal to automatically build and maintain gazetteers for named entity recognition by using wikipedia. In *Proceedings of the Workshop on NEW TEXT Wikis and blogs and other dynamic text sources*, 2006.

Kentaro Torisawa et al. Exploiting wikipedia as external knowledge for named entity recognition. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 698–707, 2007.

Shyam Upadhyay, Kai-Wei Chang, Matt Taddy, Adam Kalai, and James Zou. Beyond bilingual: Multi-sense word embeddings using multilingual context. *arXiv preprint arXiv:1706.08160*, 2017.

Lyan Verwimp, Joris Pelemans, Patrick Wambacq, et al. Character-word lstm language models. *arXiv preprint arXiv:1704.02813*, 2017.

Soroush Vosoughi, Prashanth Vijayaraghavan, and Deb Roy. Tweet2vec: Learning tweet embeddings using character-level cnn-lstm encoder-decoder. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 1041–1044, 2016.

Zhilin Yang, Ruslan Salakhutdinov, and William Cohen. Multi-task cross-lingual sequence tagging from scratch. *arXiv preprint arXiv:1603.06270*, 2016.

Zhilin Yang, Ruslan Salakhutdinov, and William W Cohen. Transfer learning for sequence tagging with hierarchical recurrent networks. *arXiv preprint arXiv:1703.06345*, 2017.

Roman Yangarber, Winston Lin, and Ralph Grishman. Unsupervised learning of generalized names. In *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002.

Sheryl R Young and Philip J Hayes. Automatic classification and summarization of banking telexes. In *CAIA*, pages 402–409, 1985.

Rui Zhao, Dongzhe Wang, Ruqiang Yan, Kezhi Mao, Fei Shen, and Jinjiang Wang. Machine health monitoring using local feature-based gated recurrent unit networks. *IEEE Transactions on Industrial Electronics*, 65(2):1539–1548, 2017.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. Deep learning for chinese word segmentation and pos tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 647–657, 2013.