An Attention-RNN based approach for Named Entity Disambiguation with Noisy Texts

Yotam Eshel
An Attention-RNN based approach for Named Entity Disambiguation with Noisy Texts

Research Thesis

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

Yotam Eshel

Submitted to the Senate of the Technion — Israel Institute of Technology
Tamuz 5778 Haifa July 2018
This research was carried out under the supervision of Prof. Shaul Markovitch, in the Faculty of Computer Science.

Some results in this thesis have been published as articles by the author and research collaborators in conferences and journals during the course of the author’s masters research period, the most up-to-date versions of which being:


The generous financial help of the Technion is gratefully acknowledged.
Contents

List of Figures

Abstract 1

1 Introduction 3

2 Disambiguation: The problem of noisy text 5
  2.1 Overview of named entity disambiguation 5
  2.2 The problem with noisy text 6

3 RNN and attention based algorithm for disambiguation of noisy text 9
  3.1 RNN overview 9
  3.2 Our Attention-RNN model for NED 10
  3.3 Training 12
  3.4 Obtaining word and entity embeddings for model initialization 13
    3.4.1 Skip-Gram based model 13
    3.4.2 Setup for training our embeddings 14

4 Evaluation 15
  4.1 Building a noisy NED dataset 15
  4.2 Evaluation on WikilinksNED 16
    4.2.1 Evaluation procedure 16
    4.2.2 Results 17
  4.3 Evaluation on CoNLL-YAGO 18
    4.3.1 Evaluation procedure 18
    4.3.2 Training our model for CoNLL-YAGO 18
    4.3.3 Results 19
  4.4 Effects of initialized embeddings and corrupt-sampling schemes 20
  4.5 Error Analysis 20

5 Related Work 23
  5.1 Local and Global NED 23
  5.2 Neural Approaches 25
List of Figures

3.1 The architecture of our Neural Network model. A close-up of the Attention-RNN component appears in the dashed box. 12
Abstract

Named entity disambiguation (NED) is the task of linking mentions of entities in text to a knowledge base, such as Freebase or Wikipedia. Currently, research on the task of NED is driven by a number of standard datasets. These datasets are based on news and encyclopedic texts that are naturally coherent, well-structured and rich. However, texts in other scenarios such as web fragments, social media or search queries are shorter, less coherent and more challenging in general.

To address the task of NED for noisy text we design a novel neural model based on RNNs and attention. Our algorithm can utilize large amounts of training samples and learn to capture the limited and noisy local context surrounding entity-mentions in noisy text. We train our model with a novel method for sampling informative negative examples. In addition, we describe a new way of initializing word and entity embeddings that significantly improves performance.

To facilitate research on NED with noisy text, we present WikilinksNED: A large-scale NED dataset of text fragments from the web that is based on the Wikilinks dataset. Our dataset is orders of magnitude larger, significantly noisier and more challenging than existing news-based datasets.

We evaluate our model both on WikilinksNED and a smaller newswire dataset and find our model significantly outperforms existing state-of-the-art methods on WikilinksNED while achieving comparable performance on the smaller dataset.
Chapter 1

Introduction

Named Entity Disambiguation (NED) is the task of linking mentions of entities in text to a given knowledge base, such as Freebase or Wikipedia. NED is a key component in Entity Linking (EL) systems, focusing on the disambiguation task itself, independently from the tasks of Named Entity Recognition (detecting mention bounds) and Candidate Generation (retrieving the set of potential candidate entities). NED has been recognized as an important component in NLP tasks such as semantic parsing [BL14].

Current NED research is mostly driven by a number of standard datasets, such as CoNLL-YAGO [HYB+11], TAC KBP [JGD+10] and ACE [BFG+10]. These datasets are based on news corpora and Wikipedia, which are naturally coherent, well-structured, and rich in context. Global disambiguation models [GB14, PHG15, GLC+16] leverage this coherency by jointly disambiguating all the mentions in a single document. However, domains such as web-page fragments, social media, or search queries, are often short, noisy, and less coherent; such domains lack the necessary contextual information for global methods to pay off, and present a more challenging setting in general.

In this work, we investigate the task of NED in a setting where only local and noisy context is available. In particular, we create a dataset of 3.2M short text fragments extracted from web pages, each containing a mention of a named entity. Our dataset is far larger than previously collected datasets, and contains 18K unique mentions linking to over 100K unique entities. We have empirically found it to be noisier and more challenging than existing datasets. For example:

“I had no choice but to experiment with other indoor games. I was born in Atlantic City so the obvious next choice was Monopoly. I played until I became a successful Captain of Industry.”

This short fragment is considerably less structured and with a more personal tone than a typical news article. It references the entity Monopoly (Game), however expressions such as “experiment” and “Industry” can distract a naive disambiguation model because they are also related to the much more common entity Monopoly (economics term). Some sense of local semantics must be considered in order to separate the useful signals
(e.g. “indoor games”, “played”) from the noisy ones.

We therefore propose a new model that leverages local contextual information to disambiguate entities. Our neural approach (based on RNNs with attention) leverages the vast amount of training data in WikilinksNED to learn representations for entity and context, allowing it to extract signals from noisy and unexpected context patterns.

While convolutional neural networks [SLT+15, FLDK16] and probabilistic attention [LSRP15] have been applied to the task, this is the first model to use RNNs and a neural attention model for NED. RNNs account for the sequential nature of textual context while the attention model is applied to reduce the impact of noise in the text.

Our experiments show that our model significantly outperforms existing state-of-the-art NED algorithms on WikilinksNED, suggesting that RNNs with attention are able to model short and noisy context better than current approaches. In addition, we evaluate our algorithm on CoNLL-YAGO [HYB+11], a dataset of annotated news articles. We use a simple domain adaptation technique since CoNLL-YAGO lacks a large enough training set for our model, and achieve comparable results to other state-of-the-art methods. These experiments highlight the difference between the two datasets, indicating that our NED benchmark is substantially more challenging.

Code and data used for our experiments can be found at https://github.com/yotam-happy/NEDforNoisyText
Chapter 2

Disambiguation: The problem of noisy text

In this chapter we give a general overview of the task of NED, discuss the problem of NED with noisy texts and present a novel dataset that facilitates research on NED with noisy text.

2.1 Overview of named entity disambiguation

Named entity disambiguation (NED) is the task of automatically choosing the correct named entity mentioned by a word or expression (a mention) within a document, from a given knowledge-base of entities. It is a core component in entity linking (EL) where mentions are identified in a given document, a limited yet high recall set of candidate entities is collected, and the NED algorithm is employed to decide on the single correct entity. In this work we isolate and focus on the NED algorithm alone, by assuming the mentions are identified and candidate entities are given.

Identifying mentions and collecting candidates has been discussed in a number of previous works and usually relies on building a large dictionary of possible (mention, entity) pairs from the given knowledge base (See [SWH15] for a survey). If Wikipedia is used as a knowledge base (as is done in this work) this dictionary can be mined from cross-article links, disambiguation pages and alternate page titles.

A knowledge-base of named entities is an important part of an NED system since it determines the coverage of known entities and provides important background information on them. In this work we employ Wikipedia since with over 5 million articles, 1 billion words and over 80 million cross-articles links (as of 2017), it has a wealth of context for disambiguation and a high coverage of entities across many domains. We follow common practice and assume each Wikipedia article represents a single entity. It should be noted however that Wikipedia contains words and abstract entities (e.g. love, table, walking) as well as named entities, thus somewhat blurring the meaning of named entity disambiguation. It seems that the research community has de-facto
decided that Wikipedia’s advantages outweigh this issue.

Another task that is sometimes addressed as part of NED is NIL-detection, where in some tasks (all-words NED) the system should detect mentions that refer to out-of-vocabulary entities. A simple method for this task is thresholding, where entity assignments must have a confidence above some threshold - otherwise NIL is returned [BP06, SWH15]. Other methods either add a NIL candidate to the set of candidates or add another step after disambiguation that discards unreliable links [RRDA11, SWH15]. In this work we do not consider NIL-detection, but focus only on ranking possible entity candidates given a mention.

2.2 The problem with noisy text

NED algorithms generally rely on three types of features for disambiguation: entity commonness, mention surface string, and textual context. Out of these, the textual context captures the essence of the disambiguation algorithm - choosing the correct entity given the surrounding context. However, context features are naturally susceptible to the quality of the text they model.

Currently, NED algorithms are usually trained and evaluated on a number of newswire-based evaluation datasets, such as TAC and CoNLL, and knowledge-bases such as Wikipedia and DBPedia. These resources contain text that is generally well edited, well formed, rich in context, and coherent. However, this evaluation might misrepresent many domains such as web fragments, tweets and search engine queries, where text is often shorter, less formal and less well written. These domains of noisy text are challenging for NED algorithms, since these rely on extracting contextual signals for disambiguation.

For example observe the following fragment taken from the web, with a much looser language than most news articles:

"We almost looked into lyme, but i tested positive for the horse (mononucleosis) so there was no need to look for a zebra. By October I had only minimal improvement ... It was time to go zebra hunting!"

This fragment refers to the medical term zebra which means a rare diagnosis when a simpler and commonplace one is more likely. In this fragment the usage of slang terms (including the mention of interest itself) can act as noise to a disambiguation algorithm that should pick Zebra (Medical) over the much more common Zebra the animal.

Another form of noise is incoherent text, such as in lists or fragmented text. In these cases the relevant signals for disambiguation are few, relative to the size of the text. For example this fragment that is taken from the web, of semi-famous individuals that died from a medical problem:

"Jerald Terhorst, 87, american white-house secretary (1974), heart failure."
Shirley mills, 83, american actress *(The Grapes of Wrath)*, pneumonia. Paul fry, 45, british motorcyclist...

The algorithm should pick *The_Grapes_of_Wrath_(film)* over the much more common *The_Grapes_of_Wrath_(novel)* while ignoring the large amount of completely unrelated signals that can generate a lot of noise.

Another example of perceived incoherence is cases where the same sentence refers to two entities of the same mention, while the mention refers only to one. For example:

"The oldest living tree in the world is a white mountain, california, bristlecone pine named *Methuselah*, after the biblical figure who lived to 969 years old"

"His best selling novels ”american gods” and ”anansi boys” helped establish his credentials with the critics, and the sly 1998 fantasy *Stardust* was adapted to the screen in 2007. His other Hollywood pursuits..."

The first part of the first fragment refers to the correct entity *Methuselah_(tree)* while the second part to the incorrect entity *Methuselah* the biblical figure. The second fragment has two parts as well: it refers to a list of novels in its first half, and the correct entity *Stardust_(novel)* among them, while the second half refers to movies and can mislead a disambiguation algorithm to choose the more common *Stardust_(film)*. Some notion of semantics is important for cases such as these in order to correctly choose the right entity.

These examples are all taken from a novel web-based and noisy NED dataset that we present in section 4.1. They highlight some types of noise that are found when dealing with web-based text, and the difficulty of designing features to capture such text. Due to this difficulty, we leverage a deep-learning approach to this problem and make use of a large training dataset to automatically learn to extract useful features from noisy and difficult to model context patterns.
Chapter 3

RNN and attention based algorithm for disambiguation of noisy text

In this section we describe our RNN and attention based approach to solving NED with noisy text. We first give a short overview of RNNs, then we describe our model and how it can be trained, and lastly we describe an algorithm for pre-training word and entity embeddings that are used to initialize the embedding dictionaries of our model.

3.1 RNN overview

Deep neural networks are machine learning models characterized by learning how to apply a series of linear and non linear transformation on their input in order to extract useful feature representation that are used for predicting a target variable. The building blocks of a deep neural networks are layers, where a simple feed forward layer is:

\[ x_{\text{out}} = f(Wx_{\text{in}}) \]

here \( W \) is a matrix and \( f \) is some non-linear function (e.g. sigmoid). In a deep feed-forward network, a composition of such transformations is applied on the input. The matrix \( A \) is the trainable parametrization of a layer, and the entire network can be trained to minimize some loss function using stochastic gradient descent (or some variant of) and the backpropogation algorithm. Deep neural networks can be seen as hierarchical feature extraction models where as part of training the network to minimize a given loss function, each layer is trained to extract more relevant and high level features based on the feature output of the previous layer.

Recurrent neural networks (RNNs) are a family of deep neural models that are designed to process sequential data such as time series and text. The simplest form of a
The recurrent layer is:

\[ h_t = f(Wx_t + Uh_{t-1}) \]  \hspace{1cm} (3.2)

Here \( x_t \) and \( h_t \) are input and output vectors at timestep \( t \) and \( h_{t-1} \) is the output vector of the network at timestep \( t - 1 \). The recurrent layer combines the vector representation of the head of the sequence \( x_1, \ldots, x_{t-1} \) with the next point in the sequence \( x_t \). In such a way, iteratively applying the recurrent layer to the sequence produces a fixed size vector representation of the entire sequence. When used for classification, the output of the RNN is usually fed into one or more feed-forward layers and then used for predicting a target variable. Similarly to feed forward networks, RNNs are trained to minimize some loss function using a variant of stochastic gradient descent.

A more complex class of RNNs are gated RNNs. Gated RNNs use a number of gates to control data flow through the network. In this work we use Gated Recurrent Units (GRUs) [CvMG+14], however other variants exist (namely LSTMs [HS97]). A GRU has two gates: an update gate \( z \) and a reset gate \( r \). Both gates compute vectors that are then used to control the data flow:

\[ z_t = \sigma(W_z x_t + U_z h_{t-1}) \]
\[ r_t = \sigma(W_r x_t + U_r h_{t-1}) \]  \hspace{1cm} (3.3)

Here \( x_t \) is again the input at timestep \( t \) and \( h_{t-1} \) is the output of the GRU at timestep \( t - 1 \). To compute the output at step \( t \) first a candidate new state \( \tilde{h}_t \) is computed by:

\[ \tilde{h}_t = \tanh(Wx_t + U(r_t \odot h_{t-1})) \]  \hspace{1cm} (3.4)

This combines the last output with new input, but first allows the network to ‘reset’ some of the last output using the output of the reset gate. Here \( \odot \) denotes point-wise multiplication. Finally the output is an interpolation between the last state and the newly computed state, controlled by the update gate:

\[ h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \]  \hspace{1cm} (3.5)

The complex gated RNNs have been constructed to address the vanishing gradient problem, shown to be beneficial for learning longer term relationship in the sequence, and demonstrated to perform well in practice [CGCB14].

### 3.2 Our Attention-RNN model for NED

To solve our disambiguation problem we design a novel RNN-based deep neural network, since deep neural networks can efficiently utilize the large amounts of available training data for extracting useful context patterns. Our deep neural model is a discriminative
model which takes a pair of local context and a candidate entity, and outputs a probability-like score for the candidate entity being a correct assignment. Both words and entities are represented using embedding dictionaries and we interpret local context as a window-of-words to the left and right of a mention. The left and right contexts are fed into a duo of Attention-RNN components which process each side and produce a fixed length vector representation. The resulting vectors are concatenated and along with the entity embedding are then fed into a classifier network with two output units that are trained to emit a probability-like score of the candidate being a correct or corrupt assignment.

Following is a detailed description of the components of our model:

**Embedding:** The embedding layer first embeds both the entity and the context words as vectors with 300 dimensions each.

**Attention-RNN:** The Attention-RNN unit is composed of an RNN and an attention mechanism. An RNN, as explained earlier, scans through the input text while maintaining an internal state allowing it to “remember” important signals in the context and to recognize signals spanning multiple words. Our implementation uses a GRU unit \([CvMG+14]\) as an RNN. We fit the GRU unit with an additional attention mechanism, commonly used with state-of-the-art encoder-decoder models \([BCB14, XBK+15]\). Since our model lacks a decoder, we use the entity embedding as a control signal for the attention mechanism. Equation 3.6 details the equations governing the attention model:

\[
a_t \in \mathbb{R}; a_t = r_{\Theta_3}(o_t, v_{candidate})
\]

\[
a_t' = \frac{1}{\sum_{i=1}^{t} \exp\{a_i\}} \exp\{a_t\}
\]

\[
o_{\text{attn}} = \sum_{t} a'_t o_t
\]  

(3.6)

The function \(r\) computes an attention value for each word processed by the RNN, using the RNN output \(o_t\) and the candidate entity \(v_{candidate}\). The final output vector \(o_{\text{attn}}\) is the sum of all the output vectors of the RNN weighted according to the attention values. This allows the attention mechanism to decide on the importance of different context parts when examining a specific candidate. We parametrize the attention function \(r\) as a single layer NN as shown in equation 3.7, similarly to the parametrization of Bahdanau et al. \([BCB14]\).

\[
r_{\Theta_3}(o_t, v_{candidate}) = Ao_t + Bv_{candidate} + b
\]  

(3.7)

**Classifier:** The classifier network consists of a hidden layer and an output layer with two output units in a softmax. The hidden layer is 300-dimensional with a ReLU
activation function, and $p = 0.5$ dropout is applied as a regularizer. The output units are trained by optimizing a cross-entropy loss function.

...indoor games. I was born in Atlantic City so the obvious next choice was Monopoly. I played until I became a successful Capitain of Industry..."

![Diagram of the architecture of the Neural Network model.](image)

Figure 3.1: The architecture of our Neural Network model. A close-up of the Attention-RNN component appears in the dashed box.

### 3.3 Training

We assume our model is only given training examples of correct entity assignments and therefore use corrupt-sampling, where we automatically generate examples of wrong assignments. For each context-entity pair $(c, e)$, where $e$ is the correct assignment for $c$, we produce $k$ corrupt samples with the same context $c$ but with a different, corrupt entity $e'$. We considered two alternatives for corrupt sampling and provide an empirical comparison of the two approaches (See 4.4):

**Near-Misses:** Sampling out of the candidate set of each mention. We have found this to be more effective where the training data reliably reflects the test-set distribution.

**All-Entity:** Sampling from the entire dictionary of entities. Better suited to cases where the training data or candidate generation does not reflect the test-set well. Has an added benefit of allowing us to utilize unambiguous training examples where only a single candidate is found.
We sample corrupt examples uniformly in both alternatives since with uniform sampling the ratio between the number of positive and negative samples of an entity is higher for popular entities, thus biasing the network towards popular entities. In the All-Entity case, this ratio is approximately proportional to the prior probability of the entity. Preliminary experiments revealed that corrupt-sampling according to the distribution of entities in the dataset (as is done by Mikolov at el. [MSC+13]), rather than uniform sampling, did not perform well in our settings due to the lack of bias toward popular entities.

Model optimization was carried out using standard backpropagation and an AdaGrad optimizer [DHS11]. We allowed the error to propagate through all parts of the network and fine tune all trainable parameters, including the word and entity embeddings themselves. We found the performance of our model substantially improves for the first few epochs and then continues to slowly converge with marginal gains, and therefore trained all models for 8 epochs with $k = 5$ for corrupt-sampling.

### 3.4 Obtaining word and entity embeddings for model initialization

Training our RNN model implicitly embeds the vocabulary of words and collection of entities in a common space since we allow the error to propagate through the embedding dictionaries. However, we found that explicitly initializing these embeddings using vectors pre-trained over a large collection of unlabeled data significantly improved performance (see Section 4.4). We therefore present a novel method for efficiently training word and entity embeddings in a common space that are later used to initialize the embedding dictionaries of our model.

#### 3.4.1 Skip-Gram based model

Our aim is to embed both words and entities in a common low dimensional space such that both a notion of word-entity relatedness and word-word semantic relations persist. To do so we modified the Skip-Gram Negative Sampling (SGNS) model proposed by Mikolov at el. [MSC+13] to train both word and entity embedding. The original Skip-gram model was designed as a distributional word embedding model which maximizes $P(c|w)$ - the probability of predicting a context word $c$ given a center word $w$, where $c$ is drawn from a window of words around $w$. This follows the *distributional hypothesis* which states that related words share similar contexts. We experimented with a modification of this objective: we consider $P(e|w)$ where $e$ are entities and $w$ are words related to these entities - words which appear in the Wikipedia article of $e$. For training we used the negative-sampling objective defined by Mikolov [MSC+13]:
\[ \log \sigma(v_v^T w) + \sum_{i=1}^{k} E_{\epsilon \sim P_\epsilon(e)} [\log \sigma(-v_v^T v_e')] \]

This formulation encourages embeddings of words relevant to similar entities to be close in the embedding space. Conversely, Levy et al. [LG14b] showed that it implicitly factorizes the word-context PPMI matrix, which means entities are as well embedded close to relevant words.

3.4.2 Setup for training our embeddings

We used word2vec\(^1\) [LG14a], which allows one to train word and context embeddings using arbitrary definitions of ”word” and ”context” by providing a dataset of word-context pairs \((w, c)\), rather than a textual corpus. To compile a dataset of \((w, e)\) pairs, we consider every word \(w\) that appeared in the Wikipedia article describing entity \(e\). We limit our vocabularies to words that appeared at least 20 times in the corpus and entities that contain at least 20 words in their articles. We ran the process for 10 epochs and produced vectors of 300 dimensions; other hyperparameters were set to their defaults.

To make sure our word embeddings maintain similar semantic regularities as the original word2vec embeddings we tested our embeddings with the analogies dataset [MCCD13]. We found our embeddings achieve 58% accuracy on the semantic part of the dataset, similar to the original results presented by Mikolov but 5% less than retraining the original word2vec model on the same Wikipedia corpus we used. On the syntactic part of the dataset our embeddings perform poorly and achieve 38% accuracy compared to 63% by word2vec. This can be expected as we train the word embeddings to predict the document rather than near-by words thus focusing on semantics and much less on syntactics. While this result leaves room for improvement, the obtained vectors do capture important semantic regularities and have proven very useful for initializing our RNN model.

\(^1\)http://bitbucket.org/yoavgo/word2vecf
Chapter 4

Evaluation

In this chapter we describe our experimental setup and present WikilinksNED, a new dataset for noisy NED. We then compare our model to the state of the art on two datasets: WikilinksNED, and the commonly-used CoNLL-YAGO dataset [HYB+11]. We also examine the effect of different corrupt-sampling schemes, and of initializing our model with pre-trained word and entity embeddings.

In all experiments, our model was trained with fixed-size left and right contexts (20 words in each side). We used a special padding symbol when the actual context was shorter than the window. Further, we filtered stopwords using NLTK’s stop-word list prior to selecting the window in order to focus on more informative words. Our model was implemented using the Keras [Cho15] and Tensorflow [AAB+15] libraries.

4.1 Building a noisy NED dataset

To facilitate our research on noisy text we first introduce WikilinksNED, a large-scale NED dataset based on text fragments from the web. Our dataset is derived from the Wikilinks corpus [SSPM12], which was constructed by crawling the web and collecting hyperlinked expressions (mentions) linking to Wikipedia articles (entities) and their surrounding text (context). Wikilinks contains 40 million mentions covering 3 million entities, collected from over 10 million web pages.

Wikilinks can be seen as a large-scale, naturally-occurring, crowd-sourced dataset where thousands of human annotators provide ground truth for mentions of interest. This means that the dataset contains various kinds of noise, especially due to incoherent contexts. The contextual noise presents an interesting test-case that supplements existing datasets that are sourced from mostly coherent and well-formed text.

To get a sense of textual noise we have set up a small experiment where we measure the similarity between entities mentioned in WikilinksNED and their surrounding context, and compare the results to CoNLL-YAGO. We use state-of-the-art word and entity embeddings obtained from [YSTT16] that are trained over a Wikipedia dump, and compute cosine similarity between embeddings of the correct entity assignment and
the mean of context words. We compare results from all mentions in CoNLL-YAGO to a sample of 50000 web fragments taken from WikilinksNED, using a window of words of size 40 around entity mentions. We find that similarity between context and correct entity is indeed lower for web mentions (0.163) than for CoNLL-YAGO mentions (0.188), and find this result to be statistically significant with very high probability ($p < 10^{-5}$). This result suggests that web fragments in WikilinksNED are indeed noisier compared to CoNLL-YAGO documents.

We prepare our dataset from the local-context version of Wikilinks\(^1\), and resolve ground-truth links using a Wikipedia dump from April 2016\(^2\). We use the page and redirect tables for resolution, and keep the database pageid column as a unique identifier for Wikipedia entities. We discard mentions where the ground-truth could not be resolved (only 3\% of mentions).

We collect all pairs of mention $m$ and entity $e$ appearing in the dataset, and compute the number of times $m$ refers to $e$ ($\#(m,e)$), as well as the conditional probability of $e$ given $m$: $P(e|m) = \#(m,e)/\sum_{e'} \#(m,e')$. Examining these distributions reveals many mentions belong to two extremes – either they have very little ambiguity, or they appear in the dataset only a handful of times and refer to different entities only a couple of times each. We deem the former to be less interesting for the purpose of NED, and suspect the latter to be noise with high probability. To filter these cases, we keep only mentions for which at least two different entities have 10 mentions each ($\#(m,e) \geq 10$) and consist of at least 10\% of occurrences ($P(e|m) \geq 0.1$). This procedure aggressively filters our dataset and we are left with 3.2M mentions.

Finally, we randomly split the data into train (80\%), validation (10\%), and test (10\%), according to website domains in order to minimize lexical memorization [LRBD15].

### 4.2 Evaluation on WikilinksNED

We describe our evaluation procedure on WikilinksNED and present our results.

#### 4.2.1 Evaluation procedure

When evaluating on WikilinksNED we use the training set for training and report evaluation of all systems on the testing set. When training our model, we use Near-Misses corrupt-sampling, which was found to perform well due to a large training set that represents the test set well.

To isolate the effect of candidate generation algorithms, we used the following simple method for all systems: given a mention $m$, consider all candidate entities $e$ that appeared as the ground-truth entity for $m$ at least once in the training corpus. This simple method yields 97\% ground-truth recall on the test set.

---

\(^1\)http://www.iesl.cs.umass.edu/data/wiki-links

\(^2\)https://dumps.wikimedia.org/

16
Table 4.1: Evaluation on noisy web data (WikilinksNED)

<table>
<thead>
<tr>
<th>Model</th>
<th>Sampled Test Set (10K)</th>
<th>Full Test Set (300K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline (MPS)</td>
<td>60</td>
<td>59.6</td>
</tr>
<tr>
<td>Cheng (2013)</td>
<td>50.7</td>
<td>-</td>
</tr>
<tr>
<td>Yamada (2016)</td>
<td>67.6</td>
<td>66.9</td>
</tr>
<tr>
<td>Our Attention-RNN</td>
<td>73.2</td>
<td>73</td>
</tr>
<tr>
<td>Our RNN, w/o Attention</td>
<td>72.1</td>
<td>72.2</td>
</tr>
</tbody>
</table>

Since we are the first to evaluate NED algorithms on WikilinksNED, we ran a selection of existing local NED systems and compared their performance to our algorithm’s:

- **Yamada et al.** [YSTT16] created a state-of-the-art NED system that models entity-context similarity with word and entity embeddings trained using the skip-gram model. We obtained the original embeddings from the authors, and trained the statistical features and ranking model on the WikilinksNED training set. Our configuration of Yamada et al.’s model used only their local features.

- **Cheng et al.** [CR13] have made their global NED system publicly available. This algorithm uses GLOW [RRDA11] for local disambiguation. We compare our results to the ranking step of the algorithm, without the global component. Due to the long running time of this system, we only evaluated their method on the smaller test set, which contains 10,000 randomly sampled instances from the full 320,000-example test set.

- Finally, we include the **Most Probable Sense (MPS)** baseline, which selects the entity that was seen most with the given mention during training.

### 4.2.2 Results

We used standard micro P@1 accuracy for evaluation. Experimental results comparing our model with the baselines are reported in Table 4.1. Our RNN model significantly outperforms Yamada et al. on this data by over 5 points, indicating that the more expressive RNNs are indeed beneficial for this task. We find that the method by Cheng et al. performs poorly on this dataset. We believe this is due to their training procedure utilizing a fixed, specially constructed training set rather than the training set of the particular dataset evaluated. This is found to be suboptimal for WikilinksNED compared to results reported on standard news-based datasets. We as well find that the attention mechanism further improves our results by a small, yet statistically significant, margin.

https://cogcomp.cs.illinois.edu/page/software_view/Wikifier
4.3 Evaluation on CoNLL-YAGO

CoNLL-YAGO [HYB+11] is a standard dataset for evaluating NED algorithms using text from newswire articles. It was built by annotating ground-truth entities of the English CoNLL 2003 NER dataset, which in turn is based on a collection of 1393 newswire articles collected from the Reuters Corpus. We describe our evaluation procedure, how we trained our model for this dataset and present our results.

4.3.1 Evaluation procedure

For comparability with existing methods we tested our model using two publicly available candidates datasets: (1) PPRforNED - Pershina at el. [PHG15]; (2) YAGO - Hoffart at el. [HYB+11].

As a baseline we took the standard Most Probable Sense (MPS) prediction, which selects the entity that was seen most with the given mention during training. We also compare to the following papers - Francis-Landau et al. [FLDK16], Yamada at el. [YSTT16], and Chisholm et al. [CH15], as they are all strong local approaches and a good source for comparison.

4.3.2 Training our model for CoNLL-YAGO

CoNLL-YAGO has a training set with 18505 non-NIL mentions, which our experiments showed is not sufficient to train our model on. To fit our model to this dataset we first used a simple domain adaptation technique and then incorporated a number of basic statistical and string based features.

We used a simple domain adaptation technique where we first trained our model on an available large corpus of labeled data derived from Wikipedia, and then continued training the resulting model on the smaller training set of CoNLL-YAGO [MMY+16]. The Wikipedia corpus was built by extracting all hyperlinks between Wikipedia pages along with their context, resulting in over 80 million training examples. We trained our model with All-Entity corrupt sampling for 1 epoch on this data. The resulting model was then adapted to CoNLL-YAGO by training 1 epoch on CoNLL-YAGO’s training set, where corrupt examples were produced by considering all possible candidates for each mention as corrupt-samples (Near-Misses corrupt sampling).

We proceeded to use the model in a similar setting to Yamada at el. [YSTT16] where a Gradient Boosting Regression Tree (GBRT) [Fri01] model was trained with our model’s prediction as a feature along with a number of statistical and string based features defined by Yamada. The statistical features include entity prior probability, conditional probability, number of candidates for the given mention and maximum conditional probability of the entity in the document. The string based features include edit distance between mention and entity title and two boolean features indicating whether the entity title starts or ends with the mention and vice versa. The GBRT
model hyper-parameters where set to the values reported as optimal by Yamada et al.\textsuperscript{4}.

### 4.3.3 Results

Table 4.2 displays the micro and macro P@1 scores on CoNLL-YAGO test-b for the different training steps. We find that when using only the training set of CoNLL-YAGO our model is under-trained and that the domain adaptation significantly boosts performance. We find that incorporating extra statistical and string features yields a small extra improvement in performance.

<table>
<thead>
<tr>
<th>CoNLL-YAGO test-b - Training Steps Eval</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>PPRforNED</td>
</tr>
<tr>
<td>CoNLL training set</td>
</tr>
<tr>
<td>+ domain adaptation</td>
</tr>
<tr>
<td>+ GBRT</td>
</tr>
<tr>
<td>Yago</td>
</tr>
<tr>
<td>CoNLL training set</td>
</tr>
<tr>
<td>+ domain adaptation</td>
</tr>
<tr>
<td>+ GBRT</td>
</tr>
</tbody>
</table>

Table 4.2: Evaluation of training steps on CoNLL-YAGO.

The final micro and macro P@1 scores on CoNLL-YAGO test-b are displayed in table 4.3. On this dataset our model achieves comparable results, however it does not outperform the state-of-the-art, probably because of the relatively small training set and our reliance on domain adaptation.

<table>
<thead>
<tr>
<th>CoNLL-YAGO test-b (Local methods)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>PPRforNED</td>
</tr>
<tr>
<td>Our ARNN + GBRT</td>
</tr>
<tr>
<td>Yamada (2016) local</td>
</tr>
<tr>
<td>Yago</td>
</tr>
<tr>
<td>Our ARNN + GBRT</td>
</tr>
<tr>
<td>Yamada (2016) local</td>
</tr>
<tr>
<td>Francis-Landau (2016)</td>
</tr>
<tr>
<td>Chisholm (2015) local</td>
</tr>
</tbody>
</table>

Table 4.3: Evaluation on CoNLL-YAGO.

\textsuperscript{4}Learning rate of 0.02; maximal tree depth of 4; 10,000 trees.
4.4 Effects of initialized embeddings and corrupt-sampling schemes

We performed a study of the effects of using pre-initialized embeddings for our model, and of using either All-Entity or Near-Misses corrupt-sampling. The evaluation was done on a 10% sample of the evaluation set of the WikilinksNED corpus and can be seen in Table 4.4.

We have found that using pre-initialized embeddings results in significant performance gains, due to the better starting point. We have also found that using Near-Misses, our model achieves significantly improved performance. We attribute this difference to the more efficient nature of training with near misses. Both these results were found to be statistically significant.

<table>
<thead>
<tr>
<th>Wikilinks Evaluation-Set Model</th>
<th>Micro accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Near-misses, with init.</td>
<td>72.5</td>
</tr>
<tr>
<td>Near-misses, random init.</td>
<td>67.2</td>
</tr>
<tr>
<td>All-Entity, with init.</td>
<td>70</td>
</tr>
<tr>
<td>All-Entity, random init.</td>
<td>67.1</td>
</tr>
</tbody>
</table>

Table 4.4: Corrupt-sampling and Initialization

4.5 Error Analysis

We randomly sampled and manually analyzed 200 cases of prediction errors made by our model. This set was obtained from WikilinksNED’s validation set that was not used for training.

Working with crowd-sourced data, we expected some errors to result from noise in the ground truths themselves. Indeed, we found that 19.5% (39/200) of the errors were not false, out of which 5% (2) where wrong labels, 33% (13) were predictions with an equivalent meaning as the correct entity, and in 61.5% (24) our model suggested a more convincing solution than the original author by using specific hints from the context. In this manner, the mention ’Supreme leader’, which was contextually associated to the Iranian leader Ali Khamenei, was linked by our model with ’supreme leader of Iran’ while the ”correct” tag was the general ’supreme leader’ entity.

In addition, 15.5% (31/200) were cases where a Wikipedia disambiguation-page was either the correct or predicted entity (2.5% and 14%, respectively). We considered the rest of the 130 errors as true semantic errors, and analyzed them in-depth.

First, we noticed that in 31.5% of the true errors (41/130) our model selected an
<table>
<thead>
<tr>
<th>Error type</th>
<th>Fraction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>False errors</td>
</tr>
<tr>
<td>Not errors</td>
<td>19.5% (39/200)</td>
</tr>
<tr>
<td>- Annotation error</td>
<td>5% (2/39)</td>
</tr>
<tr>
<td>- Better suggestion</td>
<td>61.5% (24/39)</td>
</tr>
<tr>
<td>- Equivalent entities</td>
<td>33% (13/39)</td>
</tr>
<tr>
<td>Disambiguation page</td>
<td>15.5% (31/200)</td>
</tr>
<tr>
<td></td>
<td>True semantic errors</td>
</tr>
<tr>
<td>Too specific/general</td>
<td>31.5% (41/130)</td>
</tr>
<tr>
<td>'almost correct' errors</td>
<td>26% (34/130)</td>
</tr>
<tr>
<td>insufficient training</td>
<td>21.5% (28/130)</td>
</tr>
</tbody>
</table>

Table 4.5: Error distribution in 200 samples. Categories of true errors are not fully distinct.

entity that can be understood as a specific (6.5%) or general (25%) realization of the correct solution. For example, instead of predicting 'Aroma of wine' for a text on the scent and flavor of Turkish wine, the model assigned the mention 'Aroma' with the general 'Odor' entity. We observed that in 26% (34/130) of the error cases, the predicted entity had a very strong semantic relationship to the correct entity. A closer look discovered two prominent types of 'almost correct' errors occurred repeatedly in the data. The first was a film/book/theater type of error (8.4%), where the actual and the predicted entities were a different display of the same narrative. Even though having different jargon and producers, those fields share extremely similar content, which may explain why they tend to be frequently confused by the algorithm. A third (4/14) of those cases were tagged as truly ambiguous even for human reader. The second prominent type of 'almost correct' errors where differentiating between adjectives that are used to describe properties of a nation. Particularity, mentions such as 'Germanic', 'Chinese' and 'Dutch' were falsely assigned to entities that describe language instead of people, and vice versa. We observed this type of mistake in 8.4% of the errors (11/130).

Another interesting type of errors where in cases where the correct entity had insufficient training. We defined insufficient training errors as errors where the correct entity appeared less than 10 times in the training data. We saw that the model followed the MPS in 75% of these cases, showing that our model tends to follow the baseline in such cases. Further, in insufficient-training conditions, as our model tended to select too general entities in many cases (35.7%).
Chapter 5

Related Work

5.1 Local and Global NED

Named entity disambiguation has been considered both as a stand-alone task and as a part of entity linking, where named entities should both be found and disambiguated. When Wikipedia is used as an entity repository, the term Disambiguation to Wikipedia (D2W) is sometimes used [RRDA11]. Another commonly used term is Wikification [CR13, HCH+14] which poses the task as imitating the Wikipedia style of linking entities (e.g. only significant mentions, only linking first occurrence in text).

NED algorithms can roughly be divided into local and global approaches. Local approaches attempt to solve one ambiguous mention at a time, relying on statistical and textual features based solely on the considered mention, entity candidates and the textual context. Global approaches extend local approaches by attempting to solve all mentions within a single document together using some coherency model to enforce coherency of entities across a document.

Bunescu et al. [BP06] were the first to use Wikipedia as an entity repository for a disambiguation task. They recognized the utility of the wealth of information within Wikipedia for disambiguation and utilized a simple similarity measure between a Wikipedia article text and a mention’s context words for disambiguation. Wikipedia categories were employed as well to extract useful keyword features. Mihalcea et al. [MC07] combined a pure knowledge-base approach with a machine learning model that uses part-of-speech tags, surrounding words and a heuristic to determine ‘keywords’ for each entity (taken from the entire document). More recently Lazic et al. [LSRP15] proposed a probabilistic selective context model where the model assumes most features (words) come from some background distribution and are irrelevant and selects only a few relevant features for disambiguation. Recently Chang et al. [CSMA16] showed that very good results can be obtained even with a simple word expert approach where a separate classifier is learnt for each known mention string. Our work presents a local model as well, however we rely on deep models to extract contextual features while these works rely on manually designed features. These are difficult to design and can
be suboptimal, especially when the input is noisy and less structured as in web data.

While local approaches provide a hard-to-beat baseline [RRDA11], many subsequent works developed global approaches where all the mentions in a single document are disambiguated together to ensure coherent entity assignments across the document. The general form of a global disambiguation problem is:

$$\arg\max_{\{e_i\}} \left\{ \sum_i \phi(e_i, m_i) + \sum_i \sum_{j \neq i} \xi(e_i, e_j) \right\}$$

(5.1)

Where $m_i$ are the mentions in a document, $e_i$ are the corresponding entity assignments, $\phi$ is an entity-mention fitness score and $\xi$ is an entity-entity agreement score. Proposed methods for global disambiguation differ mainly by the construction of $\phi$ and $\xi$ and by the method used to approximate the non-tractable optimization problem.

Two early works suggested simple heuristics to relax the non-tractability problem: Cucerzan et al. [Cuc07] suggested maximizing the agreement between a superposition of all candidate entities of all mentions in the document to each candidate entity. While easy to solve, this approach potentially introduces noise to the process. Milne et al. [MW08] suggested to instead consider only non-ambiguous mentions and the articles describing their assigned entities. Their method avoids the noise problem but might neglect many useful signals. Ratinov et al. [RRDA11] suggested using a two step approach: a local method is first employed to obtain initial disambiguations to all mentions in a document and then entity agreement is considered using the output of the local method, and correcting its mistakes. A similar method was later used by Yamada et al. [YSTT16] to achieve state-of-the-art results.

Kulkarni et al. [KSRC09] took a more principled approach and explored optimizing the objective using hill-climbing search and LP solvers. More recently Cheng et al. [CR13] suggested considering hard and soft constraints based on extracting relations between entities. Globerson et al. [GLC+16] achieved state-of-the-art results with an attention-like mechanism for solving entity-entity agreement. Guo et al. and Pershina et al. used random walks on a knowledge graph to achieved good results for NED [GB14, PHG15] and for jointly solving entity linking and word-sense disambiguation [MRN14].

Global models can tap into highly-discriminative semantic signals (e.g. coreference and entity relatedness) that are unavailable to local methods, and have significantly outperformed the local approach on standard datasets [GB14, PHG15, GLC+16]. However, global approaches are difficult to apply in domains where only short and noisy text is available, as often occurs in social media, questions and answers, and other short web documents. For example, [HCH+14] had to collect many tweets from the same author in order to apply a global disambiguation model. Since this work focuses on disambiguating entities within short fragments of text, our algorithmic approach tries to extract as much information from the local context, without resorting to external signals.
5.2 Neural Approaches

Of special relatedness to our work are neural approaches. Neural models do not rely on engineering features but rather extract features as a part of the training process. The first neural approach for NED [HLL+13] used stacked auto-encoders to learn a similarity measure between mention-context structures and entity candidates. More recently, convolutional neural networks (CNNs) were employed for learning semantic similarity between context, mention, and candidate inputs [SLT+15, FLDK16]. Neural embedding techniques have also inspired a number of works that measure entity-context relatedness. Hu at el. [HHD+15] described a method to train entity embedding and category based metrics for KB entities and categories. [YSTT16] described a state-of-the-art method for embedding both entities and words in the same continuous space, allowing computing both context-entity and entity-entity similarity. Our method for obtaining word and entity embeddings for initializing our RNN model is similar to the model suggested by Yamada at el. yet our method is simpler, empirically faster to train, and does not rely on mining the noisy Wikipedia category structure.

In this paper, in contrast to earlier work, we train a recurrent neural network (RNN) model which, unlike CNNs and embeddings, is designed to exploit the sequential nature of text. We also combine the RNN with a neural attention mechanism, inspired by results from Lazic at el. [LSRP15] that successfully used a probabilistic attention-like model for NED.

5.3 NED Datasets

[CH15] showed that despite the noisy nature of web data, augmenting a Wikipedia-derived training dataset with web derived one from the Wikilinks corpus [SSPM12] can improve performance on standard datasets. In our work, we find noisy web data to be a unique and challenging test case for disambiguation. We therefore use Wikilinks to construct a new stand-alone disambiguation benchmark that focuses on noisy text, rather than use it for training alone. Moreover, we differ from Chisholm at el. by taking a neural approach that implicitly discovers useful signals from contexts, instead of manually crafting features.

Commonly-used benchmarks for NED systems have mostly focused on news-based corpora. CoNLL-YAGO [HYB+11] is a dataset based on Reuters, created by hand-annotating the CoNLL 2003 Named Entity Recognition task dataset with YAGO [SKW07] entities. It contains 1,393 documents split into train, development and test sets. TAC KBP 2010 [JGD+10] and ACE [BFG+10] are two more news-based NED datasets, however these are much smaller than CoNLL-YAGO. [RRDA11] used a random sample of paragraphs from Wikipedia for evaluation; however, they did not make their exact sample publicly available. Kulkarni at el. [KSRC09] created the IITB dataset, a densely annotated dataset of 109 pages taken from Google News and ESPNStar.
Our WikilinksNED dataset is substantially different from these datasets since they are all based on high-quality content from either news articles or Wikipedia, while WikilinksNED is a benchmark for noisier, less coherent, and more colloquial text. The annotation process is significantly different as well, as our dataset reflects the annotation preferences of real-world website authors. It is also significantly larger in size, being over 100 times larger than CoNLL-YAGO.

Recently, a number of Twitter-based datasets were compiled as well [MWdR12, FHS14]. These represent a much more extreme case than our dataset in terms of noise, shortness and spelling variations, and are much smaller in size. Due to the unique nature of tweets, proposed algorithms tend to be substantially different from algorithms used for other NED tasks.

5.3.1 Other related tasks

NED is very related to the task of word sense disambiguation where words are disambiguated into a thesaurus of word senses (e.g. WordNet [Mil95]). However, while word sense disambiguation deals with words (e.g. run) and senses (e.g. 'the act of running; traveling on foot at a fast pace' vs 'streak, run (an unbroken series of events)'), NED deals with mentions (e.g. Obama) and named entities (e.g. Barak Obama, former president of the united states vs Mount Obama, the highest point in Antigua and Barbuda).

Another closely related task is co-reference resolution where mentions in a document should be clustered according to the entity they refer to. The difference being that in NED we assume we have a target knowledge-base whereas in co-reference resolution the clusters don’t have to represent entities in some knowledge base (for example names of private individuals that are not famous enough to be included in some knowledge base).

Hybrid tasks exist where mentions should be linked to a knowledge-base if it contains the target entity or clustered if they refer to the same entity but it does not exist in the knowledge-base.
Chapter 6

Conclusion

In this work we addressed the problem of named entity disambiguation for noisy text. We first showed that current NED datasets are focused on relatively well edited and rich documents, while many domains such as web-page fragments can be less rich and noiseier, and present an interesting and difficult challenge. We described an attention-RNN based model that makes use of large amounts of training data and learns how to extract useful features from noisy context. We described how our model can be trained and how its embedding dictionaries can be initialized with large amounts of unlabeled data. We then presented WikilinksNED, a novel large-scale NED dataset of web-fragments. We showed that our attention-RNN model outperforms state-of-the-art models on this dataset while achieving comparable yet sub-optimal performance on a standard news-based dataset.

Our results indicated that the expressibility of attention-RNN models indeed allows us to extract useful features from noisy context, when sufficient amounts of training examples are available. This allowed our model to significantly outperform existing state-of-the-art models. We found that both using pre-initialized embedding vocabularies, and the corrupt-sampling method employed are very important for properly training our model.

The gap between results of all systems tested on both CoNLL-YAGO and WikilinksNED indicates that our dataset is indeed a challenging task. We believe it to be an important real-world scenario, that represents a distinct test-case that fills a gap between existing news-based datasets and the much noisier Twitter data [RCME11] that has received increasing attention.

Our error analysis revealed a number of possible improvements that should be addressed. Since we used the training set for candidate generation, non-sensical candidates (i.e. disambiguation pages) caused our model to err and should be removed from the candidate set. In addition, we observed that lack of sufficient training for long-tail entities is still a problem, even when a large training set is available. We believe this, and some subtle semantic cases (book/movie) can be at least partially addressed by considering semantic properties of entities, such as types and categories. We intend to address these issues in future work.
We believe this work presents a real-world scenario that complements standard NED settings of news-wire text and that RNN and attention-based models are a promising direction for this task. We hope our algorithm and dataset will inspire more work in the domain of NED in the context of web-data and of noisy text in general.
Bibliography


A million words were extracted for this document. After that, anomalies were isolated from the dataset, which contains words that appear very rarely and remain with the data for testing, validation, and verification on the datasets from which they arrived. An empirical assessment of their impact on text datasets that can be included in the dataset reveals that they are indeed more common than the words in the dataset.

A different case is the construction of a new model, based on recurrent neural networks and an attention mechanism. Recurrent neural networks are hierarchical models that are capable of learning and indexing large amounts of text data, as well as features extracted from text. These models are then used to construct a vector representation of the text for each point in the series, which are then used as input for the model.

Anomalies in this model are detected by the network of recurrent neural networks, which then compares the performance of this model with other existing methods. This model is indeed capable of understanding the nature of the problem and comparing its performance with the performance of other models on the dataset of anomalies.

As a result, the new model can be used to construct a vector representation of the text for each point in the series, which are then used as input for the model.

This model is tested on a large dataset of text, and the results show that it is capable of understanding the nature of the problem and comparing its performance with the performance of other models on the dataset of anomalies.
הباحה

הباحה נוגשת כוון תמר על ידי מසר מסקוב תצוגה תכניתי האי מובספס רובב בולב תצוגה על אלא ת pageSize מולית, תצוגה התחל כוון תצוגה תכנית בולב תצוגה של התחל כדי התחל מסר תצוגה בולב תצוגה לאל תצוגה מסר תצוגה בולב תצוגה.

כזכר ההباحה נוגשת כוון תצוגה תכניתי האי מובספס רובב בולב תצוגה על אלא תPageSize מולית, תצוגה התחל כוון תצוגה תכנית בולב תצוגה של התחל כדי התחל מסר תצוגה בולב תצוגה לאל תצוגה מסר תצוגה בולב תצוגה. להباحה נוגשת כוון תצוגה תכניתי האי מובספס רובב בולב תצוגה על אלא תPageSize מולית, תצוגה התחל כוון תצוגה תכנית בולב תצוגה של התחל כדי התחל מסר תצוגה בולב תצוגה לאל תצוגה מסר תצוגה בולב תצוגה.
המחקר בענף בגרריהות של פרופסור שאול מרקוביץ, בפקולטה למדעי המחשב.

חלק מההתוצאות בחיבור זה פרסמו מחבריהם מספר מכתביהם ובכתבי-עת במחלקה.

탁ופת מחקר המאסטר של המחבר, אשר זכה להערכה וה.VALUE של בניו:


אני מודע לטכניקות של העברת הקסדת וידיבת בישולvement.
אלגוריתמים מובוסים של חворотทะเลת
ריבוי משמעות של איזורים בטקסט רועש

חיבור על מחקר

 לשם ملي תחומי של הדרישות чаות וניהלי
מניסיון למידה במעניינית רעיוני

יוסי אשלי

ורה של התוכנית .... מוכף סטטוס לישראלי
תנוון החשע'ה חיפה יולי 2018
אלגוריתמים מובוסים לשاتهم חורשות להתרת
ריבוי משמעויות של איזוקריים בטקסט רועש

יוחם אשל