

AMERICAN UNIVERSITY OF BEIRUT

Arabic Named Entity Recognition Via Deep
Co-learning

by
Chadi Talal Helwe

A thesis
submitted in partial fulfillment of the requirements
for the degree of Master of Science
to the Department of Computer Science
of the Faculty of Arts and Sciences
at the American University of Beirut

Beirut, Lebanon
June 2017

AMERICAN UNIVERSITY OF BEIRUT

Arabic Named Entity Recognition Via Deep Co-learning

by
Chadi Talal Helwe

Approved by:

Dr. Shady Elbassuoni, Assistant Professor
Computer Science



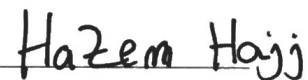
Advisor

Dr. Wassim El Hajj, Chairperson and Associate Professor
Computer Science



Member of Committee

Dr. Hazem Hajj, Associate Professor
Electrical and Computer Engineering



Member of Committee

Date of thesis defense: June 8, 2017

AMERICAN UNIVERSITY OF BEIRUT

THESIS, DISSERTATION, PROJECT RELEASE FORM

Student Name: Helwe Chadi Talal
Last First Middle

Master's Thesis Master's Project Doctoral Dissertation

I authorize the American University of Beirut to: (a) reproduce hard or electronic copies of my thesis, dissertation, or project; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes.

I authorize the American University of Beirut, to: (a) reproduce hard or electronic copies of it; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes after: **One ___ year from the date of submission of my thesis, dissertation or project.**
Two ___ years from the date of submission of my thesis, dissertation or project.
Three ___ years from the date of submission of my thesis, dissertation or project.

Chadi Helwe 22/6/2017
Signature Date

This form is signed when submitting the thesis, dissertation, or project to the University Libraries

Acknowledgements

I am deeply grateful to my thesis advisor Prof. Shady Elbassuoni for his excellent guidance, patience, and persistent help. Without him, this thesis would not have been possible. I was incredibly lucky to have him as my advisor. He taught me how to do research and how to write papers.

My sincere thanks must also go to the members of my thesis committee: Prof. Wassim El Hajj and Prof. Hazem Hajj who accepted to serve on my committee. They provided helpful suggestions to improve my work.

Special thanks must go to my University, the American University of Beirut, for those amazing two years where I met wonderful professors and new friends.

Finally, I would like deeply to thank my parents for supporting me throughout all my studies. They were always encouraging me with their best wishes.

An Abstract of the Thesis of

Chadi Talal Helwe for Master of Science
Major: Computer Science

Title: Arabic Named Entity Recognition Via Deep Co-learning

Named entity recognition (NER) is the task of identifying named entities such as locations, persons, and organizations in a given piece of text. NER plays a significant role in many applications including information retrieval, question answering, machine translation, text clustering, and navigation systems. In this thesis, we tackled the problem of Arabic NER. Arabic is a very challenging language when it comes to natural language processing (NLP) in general. Arabic is both morphologically rich and highly ambiguous and has complex morpho-syntactic agreement rules and many irregular forms. To address all these issues, we proposed to use deep learning based on Arabic word embeddings that capture syntactic and semantic relationships between words. Deep learning has been shown to perform significantly better than other approaches for various NLP tasks including NER. However, deep learning models also require a significantly large amount of training data, which is highly lacking in the case of Arabic. To be able to overcome this, we proposed a semi-supervised deep learning approach that uses both labeled and semi-labeled data, which we coin deep co-learning. We tested our approach using different established benchmarks and compared it to the state-of-the-art Arabic NER tools such as MadaMira and Farasa. Our deep co-learning approach significantly outperformed the compared to Arabic NER approaches as well as purely-supervised deep learning ones.

Contents

| | |
|--|-----------|
| Acknowledgements | v |
| Abstract | vi |
| 1 Introduction | 1 |
| 1.1 Motivation | 1 |
| 1.2 Deep Learning | 2 |
| 1.3 Word Embeddings | 3 |
| 1.4 Recurrent Neural Networks | 5 |
| 1.5 Objectives and Contributions | 5 |
| 1.6 Thesis Plan | 6 |
| 2 Arabic Challenges and Linguistic Issues | 8 |
| 3 Literature Review | 11 |
| 3.1 Rule-based Approaches | 11 |
| 3.2 Machine-learning-based Approaches | 13 |

| | | |
|----------|---|-----------|
| 3.3 | Hybrid approaches | 16 |
| 4 | Proposed Approach | 18 |
| 4.1 | Wikipedia Named Entity Recognizer | 18 |
| 4.2 | Partially Annotated Arabic NER Dataset | 22 |
| 4.3 | Deep Co-learning | 23 |
| 5 | Evaluations | 28 |
| 5.1 | Evaluation of Arabic Word Embeddings | 28 |
| 5.2 | Evaluation of Deep Co-Learning for Arabic NER | 32 |
| 5.2.1 | Validation Dataset | 33 |
| 5.2.2 | AQMAR Dataset | 34 |
| 5.2.3 | NEWS Dataset | 34 |
| 5.2.4 | TWEETS Dataset | 35 |
| 5.2.5 | Discussion | 36 |
| 5.2.6 | Error Analysis | 38 |
| 6 | Conclusion and Future Work | 40 |
| 6.1 | Conclusion | 40 |
| 6.2 | Future Work | 41 |
| A | Abbreviations | 42 |
| | Bibliography | 44 |

List of Figures

| | | |
|-----|---|----|
| 1.1 | Word Embeddings | 4 |
| 1.2 | Zoom in: Word Embeddings | 4 |
| 4.1 | LSTM + Dense Neural Network for Wikipedia NER | 19 |
| 4.2 | Automatic Labeling of Wikipedia Articles | 23 |
| 4.3 | Deep Co-learning | 24 |

List of Tables

- 5.1 Top-1 Accuracy of the different embeddings 31
- 5.2 Top-5 Accuracy of the different embeddings 32
- 5.3 Results of the models on the validation dataset 33
- 5.4 Results of the models and Arabic NER tools on the AQMAR
dataset 34
- 5.5 Results of the models and Arabic NER tools on the NEWS dataset 35
- 5.6 Results of the models and Arabic NER tools on the TWEETS
dataset 36

Chapter 1

Introduction

1.1 Motivation

Today Arabic is considered one of the most spoken languages in the world with around 420 million native speakers, however it is also one of the most difficult language when it comes to natural language processing (NLP). This has made the area of Arabic NLP a very active area of research [1]. For examples, advances have been made in various area of Arabic NLP such as POS tagging, semantic parsing, named entity recognition (NER), diacritization, tokenization, chunking, semantic role labeling (SRL), and semantically relatedness, to name a few.

In this thesis, we tackled the problem of named entity recognition for the Arabic language. We particularly focused on deep learning models for this task. Named entity recognition is an NLP task in which the goal is to extract, locate and classify the name entities in a sentence. The named entity (NE) can be

a proper noun, a numerical expression which represents type unit or monetary value, or a temporal value which represents time. The classification of a proper noun can be divided into three categories which are a person, a location, and an organization [2]. For example, in this sentence, *Washington is the capital of the USA. The country's president is Barack Obama. Washington and USA* are identified as locations and *Barack Obama* is classified as a person. Named entity recognition has a significant role in many applications such as information retrieval, question answering, machine translation, text clustering, and navigation systems [2].

While there has been many attempts to solve the problem of Arabic NER, we focused on deep learning models to achieve this in this thesis.

1.2 Deep Learning

Deep Learning is a type of machine learning that uses deep neural network architectures to learn features automatically without spending an undue effort to design them manually (i.e., feature engineering). A deep neural network is trained in an end-to-end fashion by taking raw data as input, for example a sentence or an image, which is then processed by different layers of the neural network to extract its features. The training is done by using an algorithm called backpropagation. Many researchers [3] focused on deep learning by creating models for different tasks. For example, for computer vision tasks, most of the time, a convolutional

deep learning network is trained. As For NLP tasks, a recurrent neural network is usually used [3]. Deep learning has been shown to perform significantly better than other traditional approaches on all the previously mentioned tasks. Deep learning is a very powerful type of machine learning, but it has also some disadvantages. The first drawback of using deep learning is that it needs a large amount of labeled data to generalize, which is not always publicly available. The second disadvantage is that deep learning tends to be computationally expensive.

1.3 Word Embeddings

Word embeddings are vector representations of words that are used as input features to our deep neural network classifier. Unlike n-gram representation, the benefit of the word embeddings is that the words that frequently appear in similar contexts tend to be neighbors in the embedding space. In other words, they tend to share some similar features. For example, in those two sentences “I lived in Beirut”, and “I lived in Paris”. The words Beirut and Paris share the same contextual information, which means that the two capitals are close to each other in the embedding space. Figures 1.1 and 1.2 show a visualization of the word embeddings on a 2D space.

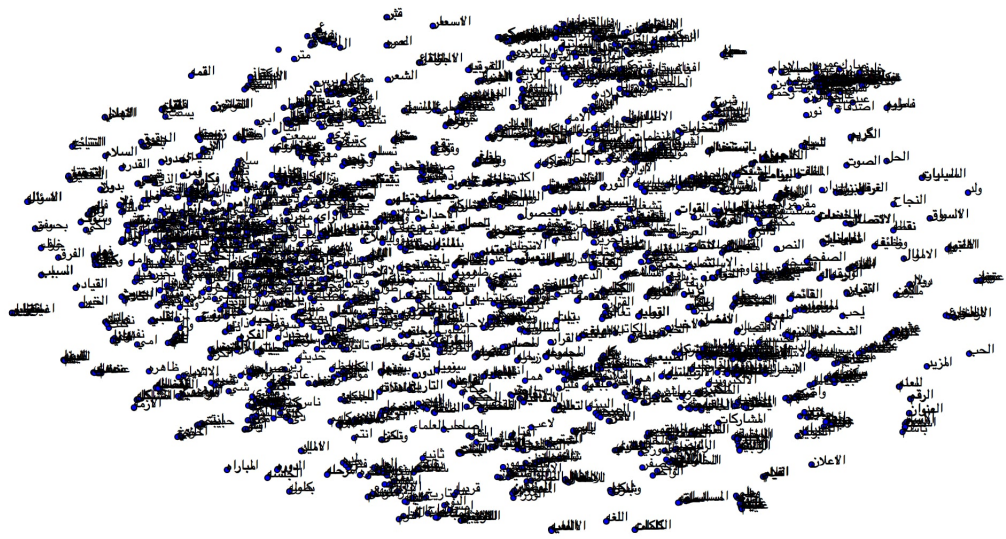


Figure 1.1: Word Embeddings

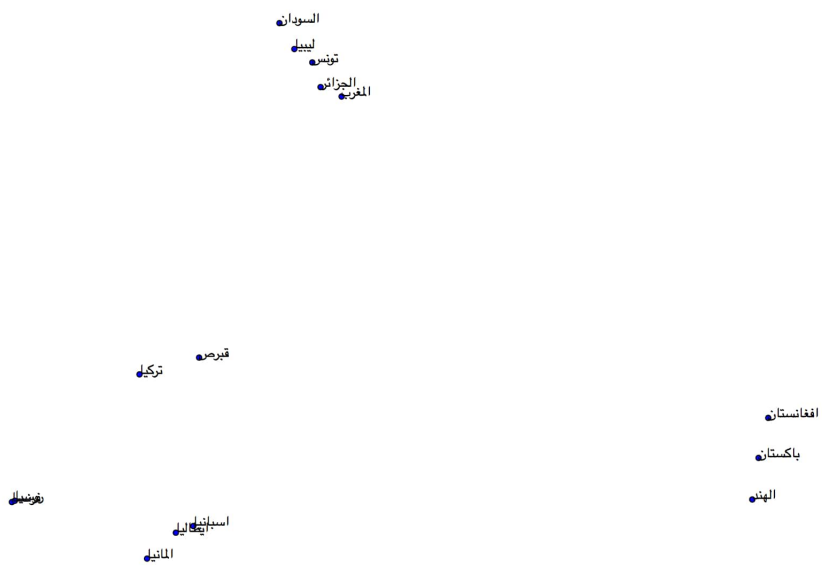


Figure 1.2: Zoom in: Word Embeddings

1.4 Recurrent Neural Networks

Recurrent neural networks (RNN) are a type of neural networks that process sequential data. Such data is in general text or stock data. RNN have some advantages comparing to other neural networks like they can process long sequence input one element at a time, such sequence can be of variable length. One specific property of a RNN is that they use parameter sharing. The advantage of parameter sharing is that it shares the weights across several time steps. RNN have an important disadvantage, which is the gradient over many steps tends to vanish. Many researchers worked on techniques to deal with this problem. One variant of RNN called Long Short-Term Memory (LSTM) proposed by Hochreiter and Schmidhuber [4] helps to keep the gradient stable. We discuss LSTM in details in Section 4.1.

1.5 Objectives and Contributions

In this thesis, we aim to develop a novel Arabic NER tool that can outperform Arabic NER tools. Our tool will be based on a deep learning approach.

As we described earlier, deep learning needs a significant amount of data to generalize. Unlike English, Arabic does not have a lot of publicly available annotated data for the task of NER. To overcome this problem we adopted co-learning, a semi-supervised algorithm, in the context of deep learning for the task of Arabic NER that can be trained on small fully annotated and large semi-annotated

datasets. We generated a large corpus from Wikipedia by partially annotating articles for the task of Arabic NER. We developed Wikipedia Named Entity Recognizer, a supervised deep learning model, that infers the type of the name entities in a Wikipedia article by classifying their Wikipedia pages into one of the four labels: person, location, organization, and other.

Our contributions in this thesis can be described as follows:

- We developed a Wikipedia Named Entity Recognizer
- We generated a partially annotated data for the task of NER from Wikipedia
- We developed a new Arabic NER tool based on a new approach called Deep Co-learning

1.6 Thesis Plan

The thesis is organized as follows: Chapter 2 presents the challenges of the Arabic language for the task of NER. Chapter 3 surveys existing work done on Arabic NER. In this chapter three approaches are discussed mainly rule-based approaches, machine-learning-based approaches, and hybrid approaches. Chapter 4 presents the proposed approach. First, we consider a classifier we developed called Wikipedia Named Entity Recognizer. Second, we described the details of the generation of a large partially annotated corpus. Finally, we present the Deep Co-learning model that is trained on labeled and semi-labeled data. Chapter 5

describes the evaluation process of the Deep Co-learning on different datasets in comparison with other models and other Arabic NER tools. Chapter 6 concludes this thesis.

Chapter 2

Arabic Challenges and Linguistic Issues

The Arabic language has many features [1, 2] that makes NER a particularly difficult task. First, the Arabic script has properties that differs a lot from the Latin script which requires the development of specific tools to be able able to process Arabic script such as the Buckwalter's Arabic Morphological Analyzer (BAMA) [5] and CJK lexical resources [6]. Another issue is the different language dialects used in Arabic, which are the classical Arabic, the Modern Standard Arabic (MSA), and the colloquial arabic dialects where each one differs from the other in the vocabulary, on the orthographic convention and the named entity form [2].

In this thesis, we focus our work on the Modern Standard Arabic. One of the advantages of the Latin script is the use of capitalization to recognize the named

entity which does not exist in the Arabic script. This problem in the Arabic script can cause ambiguity when detecting a named entity because a person name can also be an adjective. One way to resolve this issue is to look at the context surrounding the named entity [2].

In addition, the Arabic language is known for its rich morphology due to the concatenation of affixes and clitics on a stem which creates a complex word. In English, the clitics are considered as a separate word, but in Arabic, they are attached to the named entity. Two solutions are provided to solve this problem. First solution is to remove the affixes and the clitics and to keep only the root of the word [7]. Another solution that is more efficient is to use text segmentation by adding delimiters between the affixes, clitics, and root. This solution is better than the last one because we are not removing contextual information [8].

Another fundamental challenge that faces any Arabic NER system is the problem of diacritics. Diacritics affect the phonetic representation and the meaning of a word. Most of the Arabic text remove the diacritics which leads to having a different label for the same named entity. To be able to solve this problem we need to look at the surrounding contextual information of the named entity [9]. Sometimes the same named entity can be recognized as two or more different types. Two solutions were provided. The first solution is to use heuristic techniques to detect the named entity type [10], and the second solution is to develop a classifier that can predict the named entity type with the higher precision [11].

Another important issue in the Arabic script is that it suffers from the lack

of standardization. A named entity for instance can be written different ways in Arabic which leads to having variants of the named entity types. Another issue as well is dealing with spelling mistakes, which is not specific to Arabic alone.

Finally, one major issue that NLP researchers face when working on Arabic NLP tasks like the one we are concerned with here is the lack of lexical resources. Such resources includes Arabic Corpora and Arabic gazetteer for training the various NLP tools. Even if some of these resources are present, they are usually limited in scope and are not usually publicly available. In this thesis, we used deep learning and word embeddings to deal with all the linguistic issues, and we adopted a semi-supervised approach for our deep learning model to deal with the lack of training data.

Chapter 3

Literature Review

There are different types of approaches to develop an NER system. These approaches can be categorized into three main categories: rule-based approaches, machine-learning-based approaches, and hybrid approaches.

3.1 Rule-based Approaches

The rule-based approaches typically need linguistic experts to write the grammatical rules to detect the context where a named entity appears. The advantage of this method is that the NER system will be developed on a solid knowledge of grammatical rules provided by a linguistic expert. The disadvantage of this approach is how to maintain and update these rules. In a survey on Arabic NLP [2], they discussed a set of tools that were created using the rule-based approach. Such tools are TAGARAB developed by Maloney and Niv [12]. They used a

morphological analyzer to detect where the named entity begins and when the non-name context begins. They used 14 texts from the Al-Hayat CD Rom to evaluate their system. The tool had an F-measure of 85%. Abuleil [13] proposed a rule-based tool that uses lexical triggers because NEs are most probably near a lexical trigger. He used 500 texts from the Al-Raya newspaper to evaluate his approach. They achieved a precision of 90.4% on person, 93% on location and 92.3% on organization. Another strategy proposed by Samy et al. [14] is to consider two identical corpora in different languages, the first corpora is in Spanish and the second in Arabic, to perform a mapping technique between them such that each Arabic word is transliterated to Spanish and then Spanish named entities are detected and their Arabic matches are finally returned. The evaluation was performed on 1200 sentence pairs. They achieved a precision of 84% and a recall of 97.5%. The authors used a filter to remove stop words from the transliterated words. This addition improved the precision to 90%.

Mesfar [15] developed an NER system that uses a tokenizer, a morphological analyzer, and an Arabic NER, respectively. He evaluated his tool on articles from the newspaper *Le Monde Diplomatique* and achieved an F-measure of 76% for locations and an F-measure of 96% for time and numerical expression. Shaalan and Raza [16] developed an NER system called PERA that can detect the person names such as *ism*, *kunya*, *laqab*, and *nisba*. To be able to do this, they created rules using regular expressions. Also, they used gazetteers, grammar rules, and a filtering technique. They evaluated their tool on the Automatic Content Ex-

traction (ACE) [17] and Treebank Arabic dataset and obtained an F-measure of 87.5%. The same authors generalized their approach to detecting a person, a location, and an organization. They developed their corpora from the ACE, the Web, and name databases provided by organizations. They reported an F-measure of 87.7%, 85.9%, and 83.15% for persons, locations, and organizations, respectively. Elsebai et al. [18] developed a rule-based NER that can identify persons. They used heuristic rules that use verb triggers and named entity triggers, and also they used BAMA [5] to check if the word detected by the heuristic rules is a proper noun. Two evaluations were done. The first assessment was performed on 700 news articles. They achieved an F-measure of 89%. The second experiment was done on 500 articles, and resulted in an F-measure of 89%. Al-Shalabi et al. [19] developed an Arabic NER lexical trigger which used name connectors. Their approach detects different named entity types like persons, cities, localizations, and organizations. They evaluated their approach on 20 articles from the Qatari newspaper *Al-Raya* and the Jordanian newspaper *Alrai*. They reported a precision of 86.1%.

3.2 Machine-learning-based Approaches

The machine-learning-based (ML) approaches use text that contain named entities which are annotated by their types as training data. One of the early ML tools for Arabic NER is Anersys 1.0 developed by Benajiba et al. [8]. The features

used are lexical, contextual, and gazetteer features. Their first version had poor performance because it had difficulties in identifying composite named entities. They achieved an F-measure of only 55.23%. An improved version [20], Anersys 2.0, was developed which detects the boundaries of a named entity and labels the type of the identified named entities. To improve the detections of the named entity's boundaries, they used POS tagging as a feature. This system achieved an overall F-measure of 65.91%. The algorithm used is conditional random field (CRF). Benajiba and Rosso [20] employed only independent language features in Anersys 1.0. They then used in addition to those independent features, features that are Arabic dependent like POS tags, BPC, gazetteers, and nationality. They reported an F-measure of 79.21% in this case.

Another approach proposed by Benajiba et al. [21] is the use of an SVM classifier on the features of the ACE dataset which are lexical, contextual, morphological, gazetteers and syntactic features. They evaluated this approach on different ACE using cross-validation and reported an F-measure of 82.71%, 76.43%, and 81.47% for ACE 2003, ACE 2004, and ACE 2005, in this order.

Benajiba et al. [11] proposed another approach to get slightly better results by using a feature selection and a vote classifier that combines CRF and SVM. The vote classifier chooses the named entity types with the highest precision. This technique obtained an F-measure of 83.5% for ACE 2003, 76.7% for ACE 2004, and 81.31% for ACE 2005. Benajiba et al. [22] added an extra feature which is the syntagmatic feature which led to an improvement in the performance as

indicated by an F-measure of 84.32% for ACE 2003, 78.12% for ACE 2004 and 81.73% for ACE 2005.

Another approach proposed by by Abdul-Hamid and Darwish [23] is to select a set of specific features, which are boundary character n-grams, word n-gram probability-based features, word sequence features, and word length. The model used by the authors was CRF. They evaluated their system on ANERcorp [24] and reported an F-measure of 81%. Another approach proposed by AbdelRahman et al. [25] is a combination of different machine learning techniques. They applied bootstrapping, semi-supervised learning algorithm, and CFR. They used many features which are word level, POS tag, gazetteers, semantic field tag, and morphological features. They evaluated on the ANERcorp using cross validation and obtained an F-measure of 74.06% for person, 89.09% for location, 75.01% for organization, 69.47% for job, 77.52% for device, 80.95% for car, 80.63% for cellphone, 98.52% for currency, 76.99% for date, and 96.05% for time.

Koulali et al. [26] implemented a set of regular expressions to extract patterns from the text and an SVM classifier that learns from POS tagged text. The features are dependent and independent. The dependent features take into consideration the determiner used by organization name, word character features, POS features, and *verb around* features. The system was evaluated on ANERcorp and achieved an F-measure of 83.20%.

Pasha et al. [27] developed MADAMIRA a very powerful tool that is not limited to named entity recognition. Their system is based on a preprocessing phase

in which morphological features are extracted and then used as inputs to an SVM. In addition to morphological features, they used POS tagging, tokenization, and base phrase chunking as input features to the learning algorithm. Abdelali et al. [28] developed a system like MADAMIRA called FARASA. FARASA uses cross-lingual features. Also they developed their own NE gazetteer from Wikipedia. FARASA employs an SVM to label the named entities.

3.3 Hybrid approaches

Another type of approaches commonly used for NER is the hybrid approaches which combine rule-based and machine-learning-based techniques. Abdallah et al. [29] combined the rule-based approach developed by Shalaan et al. [10] and a decision tree model. They evaluated their method on ANERcorp. They reported an F-measure of 92.8% for the persons, 87.39% for locations, and 86.12% for organizations.

Oudah and Shalan [30] developed an updated version of the previous method. They extended the tool to detect and classify more NEs, they looked into two more models: SVM and Logistic Regression, and they added more features like morphological features and English capitalization feature. The last feature represents the capitalization information of an English word that was transliterated from an Arabic word. They evaluated their method on ANERcorp and obtained an F-measure of 94.4% for persons, 90.1% for locations, and 88.2% for organiza-

tions.

Chapter 4

Proposed Approach

4.1 Wikipedia Named Entity Recognizer

In this task, our goal is to classify the named entity into one of the four classes PER, LOC, ORG and OTHER. For instance, consider the article *Barack Obama* our classifier should output PER. To do this, we build a deep neural network model, which is depicted in Figure 4.1. Our model makes use of word embeddings (i.e., vector representation of words), which are learnt via deep unsupervised learning techniques such as CBOW or Skip-gram [31]. All the word vectors are stacked in a word embedding matrix $L_x \in R^{d \times |V|}$, where d is the dimension of word vector and $|V|$ is vocabulary size.

The core of our approach is a long short-term memory (LSTM) network followed by three fully-connected layers and then a final softmax layer. An LSTM network is a kind of a recurrent neural network (RNN) developed by Hochreiter

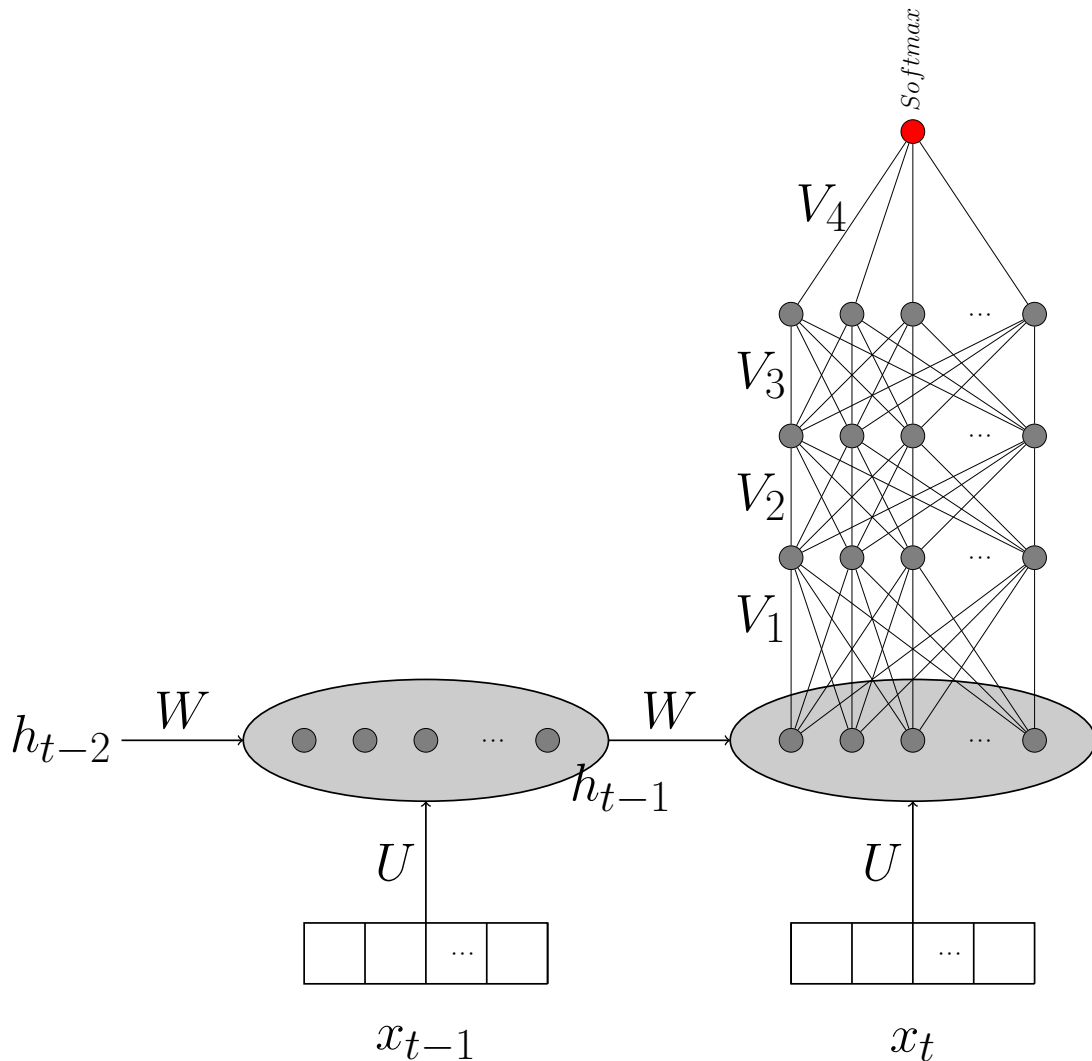


Figure 4.1: LSTM + Dense Neural Network for Wikipedia NER

and Schmidhuber [4], and is capable of mapping vectors of words with variable length to a fixed-length vector by recursively transforming the current word vector x_t with the output vector of the previous step h_{t-1} . In Figure 4.1, U , W , and V_i , where $1 \leq i \leq 4$, are weight matrices that refer to input-to-hidden, hidden-to-hidden, and hidden-to-output connections, respectively. An LSTM cell is then computed as follows [32].

$$i_t = \sigma(x_t U_i + h_{t-1} W_i + b_i) \quad (4.1)$$

$$f_t = \sigma(x_t U_f + h_{t-1} W_f + b_f) \quad (4.2)$$

$$o_t = \sigma(x_t U_o + h_{t-1} W_o + b_o) \quad (4.3)$$

$$g_t = \sigma(x_t U_g + h_{t-1} W_g + b_g) \quad (4.4)$$

$$s_t = g_t \odot i_t + s_{t-1} \odot f_t \quad (4.5)$$

$$h_t = o_t \phi(s_t) \quad (4.6)$$

i_t represents the input node. It computes the sum of the current input vector x_t , which is multiplied by the weight matrix U , and the output of the hidden layer of the previous time step h_{t-1} , which in turn is multiplied by the weight matrix W . This sum is passed to the activation function σ , which is a *sigmoid* activation function.

Similar computations are used in f_t , o_t , g_t in the above equations. f_t represents the forget gate, o_t represents the output gate and g_t represents the input gate. The forget gate f_t is used to release information from the internal state. The input gate g_t is used to add new information to the internal state. The output gate o_t is used to pass information from the internal state. s_t is the internal state of the LSTM unit, which represents the memory of the cell. It takes the sum of the input gate g_t , which is point-wise multiplied with the input node i_t , and

the forget gate f_t , which is multiplied with the previous time step of the internal state s_{t-1} . h_t represents the hidden layer, which is calculated by multiplying the output gate o_t with the \tanh function of the internal state s_t .

We also extend the LSTM network described above to create a deeper architecture by adding a dense neural network of three hidden layers and a *softmax* activation function to predict the final label of a named entity. Adding a fully connected neural network on top of an LSTM will disentangle the factors of variations in the hidden state, making it easier to predict the output. It allows the hidden state of the model to be more compact and results in the model being able to summarize the history of previous inputs more efficiently [33]. Using this architecture, the final output y_t is computed as follows.

$$y_t = \text{Softmax}(V_4\varphi(V_3\varphi(V_2\varphi(V_1h_t)))) \quad (4.7)$$

where φ represents the rectified linear unit (*ReLU*) activation function.

To be able to train the above described model, we used a set of predefined Wikipedia categories such as countries, cities, humanitarian organization, people born in certain years, etc. For each category, we retrieved the set of articles it belongs to it and assigned all these articles the suitable named entity type. For example, the articles that belong to countries and cities will be automatically assigned the type LOC.

We retrieved 12240 article summaries from Wikipedia where each category has 2720 summaries. We trained our model on 10880 summaries and validated on 1360 summaries. Our model has an F-measure of 95.4%.

4.2 Partially Annotated Arabic NER Dataset

After we had developed the Wikipedia Named Entity Recognizer, we generated a large corpus that is partially annotated. We wrote a script that uses the Wikipedia API to retrieve Wikipedia articles randomly. We partially annotated the articles by predicting the labels of the named entities that have an internal link that redirects to another Wikipedia article. For each named entity that has an internal link to a Wikipedia article, we predicted its label by using our model. The input to our model will be the summary of the article where each word is transformed into a vector representation. Figure 4.2 shows an example of how we partially annotated Wikipedia article to generate a large semi-labeled corpus for the task of Arabic NER which can be then used by our Deep Co-learning model. In Figure 4.2, our script retrieved the Arabic Wikipedia article of Steve Jobs, and then it begins to find the named entities that have an internal link, in our case, it finds two named entities which are **سان فرانسيسكو** and **حمص**. Our script will retrieve the summary of the Wikipedia articles of each NEs and then will apply the model to predict their labels. For **سان فرانسيسكو** and **حمص**, the system will assign to them a LOC label. Our approach is only limited to named entities that



Figure 4.2: Automatic Labeling of Wikipedia Articles

have internal links. For example, in the Figure 4.2, **بول وكلارا جوبز** and **عبد الفتاح** are named entities of type PER but they don't have an internal link so our tool cannot classify them. To deal with this problem, our Deep Co-learning approach will be able to predict the missing labels. Using our approach we created a corpus of 16420 sentences from 2000 different articles from Wikipedia.

4.3 Deep Co-learning

After we finally generated a large corpus, we designed our final model that predicts the type of a named entity in any given Arabic text. Our model is based on the concept of co-training proposed by Blum and Mitchell [34]. Co-training is a semi-supervised learning approach that can be trained using both labeled and unlabeled data. It uses two classifiers that learn from each other using two

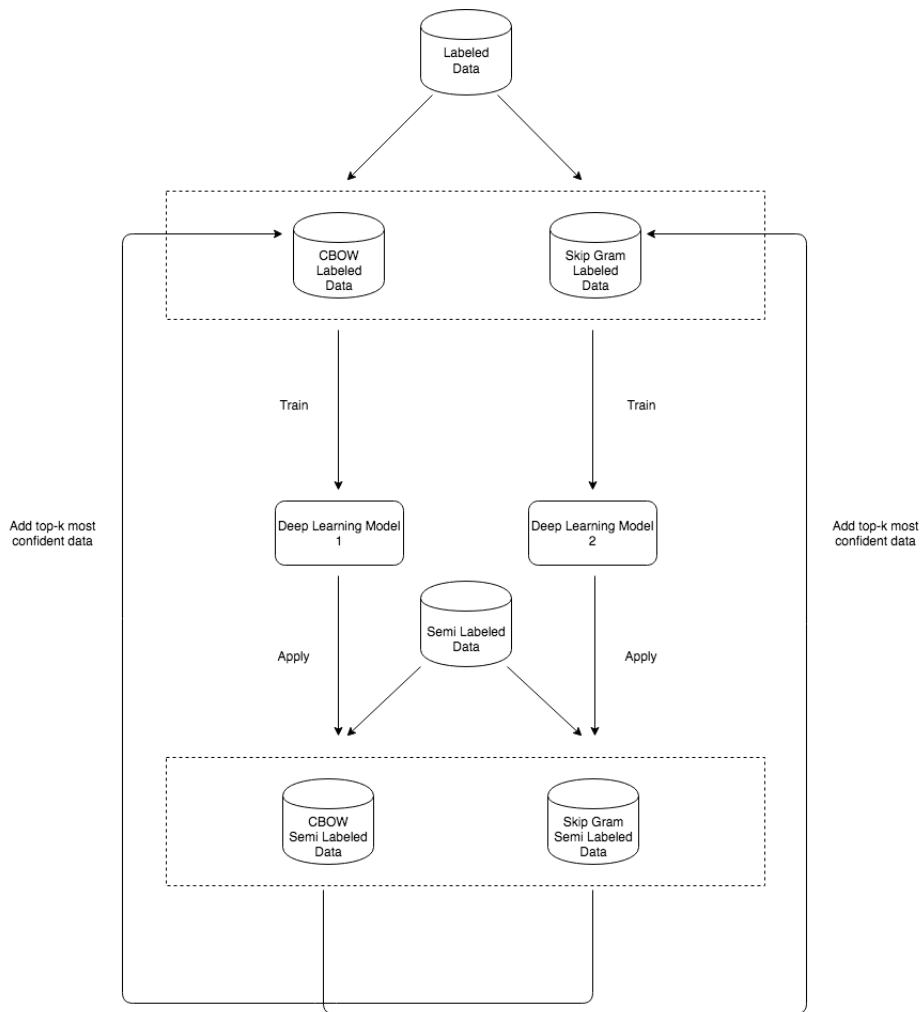


Figure 4.3: Deep Co-learning

different views of the data. For example, to classify web pages into either student or faculty pages, we can make use of two different classifiers, each based on its own set of features. For instance, the first classifier can be based on the set of words in the web page, whereas the second can be based on the set of words in the hyperlinks inside the web page. After training these two classifiers, each classifier will be used to predict the labels of unlabeled web pages. In addition, each classifier will provide a confidence level for its predictions. Finally, the set of web pages with the highest prediction confidence from each classifier will be added to the training data of the other classifier, and both classifiers will be then retrained using the augmented training data.

Our approach adapts the co-training algorithm to the realm of deep learning

Data: Labeled Data D_l , Semi-Labeled Data D_{sl} , Unlabeled Data D_{ul}

f^1 a deep learning model

f^2 a deep learning model

repeat

 Take randomly a pool $D_{sl}^{1'}$ from D_{sl}

 Take randomly a pool $D_{sl}^{2'}$ from D_{sl}

 Transform D_l^1 and $D_{sl}^{1'}$ to CB_l and CB_{sl} using CBOW

 Transform D_l^2 and $D_{sl}^{2'}$ to SG_l and SG_{sl} using Skip Gram

 Train f^1 on CB_l

 Train f^2 on SG_l

 Apply f^1 on CB_{sl}

 Apply f^2 on SG_{sl}

 Add top-k most confident predictions $TopCB_{sl}$ to D_l^2

 Add top-k most confident predictions $TopSG_{sl}$ to D_l^1

 Remove $TopCB_{sl}$ from D_{sl}

 Remove $TopSG_{sl}$ from D_{sl}

until *there are no instances in Semi-Labeled Data D_{sl} ;*

Apply f^1 and f^2 on D_{ul} by applying ensemble averaging

Algorithm 1: Deep Co-learning Algorithm

as follows. We used two different word embeddings as features. The first word embeddings we used was the Continuous Bag-of-Words (CBOW) and the second was the Skip-gram embeddings [31]. We used those two embeddings because Skip-gram is the opposite of CBOW and vice versa. CBOW predicts a word given its context, while Skip-gram predicts the context given a word. Next, we trained two deep neural network models, each based on one type of the embeddings. Initially both models was trained using a small purely-labeled dataset, namely ANERCorp. Next, each model is used to predict the label of any unlabeled data in our partially annotated Wikipedia dataset. Next, the instances with the most confident predictions by each classifier is retrieved and added to the training data of the other classifier, and the two classifiers are retrained again. We continue this process until there are no more instances left in the partially labeled dataset. Finally, we use ensemble averaging to recognize and classify named entities on unseen data. Note that the confidence per instance is computed as follows:

$$conf_instance = \frac{1}{n} \sum_{t=0}^n \arg \max_{0 \leq l \leq 7} conf(l) = \{l \mid \frac{(y_{pred_dl}^l + y_{pred_wiki}^l)}{2}\}$$

In the confidence’s formula, n represents the number of tokens in a sentence, l accounts for the label, in our case, we have seven tags which are: B-PER, I-PER, B-LOC, I-LOC, B-ORG, I-ORG, and OTHER. $y_{pred_dl}^l$ represents the probability of predicting label l using a classifier from the Deep Co-learning approach. $y_{pred_wiki}^l$ denotes the probability of predicting label l using our Wikipedia Named

Entity Recognizer. If the Wikipedia Named Entity Recognizer was not able to predict the type of a named entity because it has not an internal link, a probability of zero will be given for each l of y_{pred_wiki} . $conf(l)$ is the average of the prediction of label l between $y_{pred_dl}^l$ and $y_{pred_wiki}^l$

Our deep neural network architecture is depicted in Figure 4.3 and the pseudocode of the deep co-learning algorithm is given in Algorithm 1. Our model was trained on 80% of the ANERCorp dataset and validated and the other 20%. Then we tested on different datasets mainly: AQMAR dataset which was developed by Mohit et al. [35], NEWS and TWEETS datasets which were developed by Darwish et al [36].

Chapter 5

Evaluations

5.1 Evaluation of Arabic Word Embeddings

We evaluated different unsupervised algorithms to learn word embeddings. The evaluation was based on the word analogy task, made popular by Mikolov et al. [31], which they used to evaluate different techniques for generating English word embeddings. In a nutshell, the analogy task is composed of a set of questions formed from two pairs of words (a, b) and (c, d) as follows: "a to b is like c to?". Each such question will then be answered by calculating a target vector $t = b - a + c$. We then calculate the cosine similarity between the target vector t and the vector representation of each word w in a given word embeddings V . Finally, we retrieve the most similar word w to t , i.e.,

$$\operatorname{argmax}_{w \in V \& w \notin \{a, b, c\}} \frac{w \cdot t}{\|w\| \|t\|}$$

If $w = d$ (i.e., the same word) then we assume that the word embeddings V has answered the question correctly.

To evaluate different Arabic word embeddings, we adopted the same strategy. First, we built a word analogy benchmark specifically designed for the Arabic language. It consists of nine relations such as capitals of countries, female and male forms of words, etc. Each relation consists of over 100 word pairs. Given our benchmark, we generate a test bank consisting of over 100,000 tuples. Each tuple consists of two word pairs (a, b) and (c, d) from the same relation. Once tuples have been generated, they can be used as word analogy questions to evaluate different word embeddings as described above.

Moreover, we also extend the traditional word analogy task by taking into consideration if the correct answer is among the top-5 closest words in the embedding space to the target vector t , which allows us to more leniently evaluate the embeddings. This is particularly important in the case of Arabic since many forms of the same word exist, usually with additional prefixes or suffixes such as the equivalent of the article "the" or possessive determiners such as "her", "his", or "their". To relax this and ensure that different forms of the same word will not result in a mismatch, we use the top-5 words for evaluation rather than the top-1.

We evaluated four different Arabic word embeddings that have been generated by previous work. The first three are based on a large corpus of Arabic documents constructed by Zahran et al. [37], which consists of 2,340,895 words.

Using this corpus, the authors generated three different word embeddings using three different techniques, namely the Continuous Bag-of-Words (CBOW) model [31], the Skip-gram model [31] and GloVe [38]. The fourth word embeddings we evaluate in this thesis is the Arabic part of the Polyglot word embeddings, which was trained on the Arabic Wikipedia by Al-Rfou et al and consists of over 100,000 words [39]. To the best of our knowledge, these are the only available word embeddings that have been constructed for the Arabic language.

Table 5.1 and Table 5.2 display the accuracy of each embedding technique using the two different evaluations criteria outlined above (i.e., using top-1 and top-5 matches, respectively). Note that we consider a question to be answered wrongly if at least one of the words in the question are not present in the word embeddings. That is, we take into consideration the coverage of the embeddings as well [40].

As can be seen in Table 5.1 and Table 5.2, the CBOW model consistently outperforms all other compared models for both evaluation criteria. The performance of Polyglot is particularly low since the embeddings were trained on a much smaller corpus (Arabic portion of Wikipedia), and thus both its coverage and the quality of the embeddings are much lower.

| Relation | CBOW | Skip-gram | Glove | Polyglot |
|-------------|-------|-----------|-------|----------|
| Capital | 31% | 26.6% | 31.7% | 0.4% |
| Currency | 3.15% | 2% | 0.8% | 0.4% |
| Male-Female | 29% | 24.8% | 30.8% | 3.8% |
| Opposite | 7.6% | 4.41% | 7.3% | 2.3% |
| Comparative | 23.9% | 15.7% | 21.7% | 1.4% |
| Nationality | 29% | 29.91% | 25.8% | 0.8% |
| Past Tense | 4.3% | 2.7% | 4.5% | 0.4% |
| Plural | 23.3% | 13.28% | 19% | 2.9% |
| Pair | 8.6% | 7.6% | 1.8% | 0.02% |
| ALL | 16.3% | 12.8% | 14.5% | 1.3% |

Table 5.1: Top-1 Accuracy of the different embeddings

| Relation | CBOW | Skip-gram | Glove | Polyglot |
|-------------|--------|-----------|-------|----------|
| Capital | 42.9% | 40.8% | 47% | 1.8% |
| Currency | 4.9%% | 3.9% | 3.7% | 1.6% |
| Male-Female | 45.6% | 40.6 | 52.4% | 8.3% |
| Opposite | 15.75% | 10.65% | 19.8% | 5.4% |
| Comparative | 39.61% | 30.95% | 38.3% | 4% |
| Nationality | 34.65% | 39.6% | 32.4% | 3% |
| Past Tense | 11.4% | 9.6% | 16.7% | 1.5% |
| Plural | 45.12% | 37.9% | 41.9% | 7.2% |
| Pair | 23% | 21.3% | 5.3% | 0.07% |
| ALL | 26.6% | 23.8% | 26.4% | 3.4% |

Table 5.2: Top-5 Accuracy of the different embeddings

5.2 Evaluation of Deep Co-Learning for Arabic NER

In this study, we evaluated the potential of using Deep Co-learning for the task of Arabic NER. We tested our model on different datasets in comparison with other Arabic NER tools and models. The tools employed in this evaluation are FARASA and MADAMIRA, and the models used are supervised LSTM, supervised ensemble averaging of 2 LSTMs, Deep Co-learning of 2 LSTMs using

an unlabeled dataset, and our final model is Deep Co-learning of 2 LSTMs using a semi-labeled dataset. We trained all supervised models for 100 epochs with a batch size of 32, and we used RMSProp as the optimizer. For the Deep Co-learning models, the two deep learning models used the same configuration as of the supervised models, and we repeated the process of co-learning for 50 times.

5.2.1 Validation Dataset

As we previously mentioned, we splitted the ANERCorp dataset as 80% for training dataset and 20% for validation dataset. We validated our models on the validation dataset. Table 5.3 shows the results of different deep learning models on the validation dataset.

| Model | PER | LOC | ORG | Avg. F1 |
|--|------------|------------|------------|----------------|
| LSTM | 0.868 | 0.855 | 0.690 | 0.804 |
| Ensemble Avg. of 2 LSTMs | 0.879 | 0.870 | 0.701 | 0.816 |
| Deep Co-learning of 2 LSTMs using unlabeled data | 0.887 | 0.859 | 0.696 | 0.814 |
| Best Iteration Deep Co-learning of 2 LSTMs using unlabeled data | 0.881 | 0.872 | 0.722 | 0.825 |
| Deep Co-learning of 2 LSTMs using semi-labeled data | 0.888 | 0.873 | 0.709 | 0.823 |
| Best Iteration Deep Co-learning of 2 LSTMs using semi-labeled data | 0.893 | 0.876 | 0.721 | 0.830 |

Table 5.3: Results of the models on the validation dataset

5.2.2 AQMAR Dataset

After we had found the best models, we tested our models on the AQMAR dataset. AQMAR dataset is an annotated corpus for the task of NER. The corpus was developed by Mohit et al [35]. AQMAR contains 2456 sentences from 28 articles from Arabic Wikipedia. The articles come from 4 domains mainly: history, science, sports, and technology. Table 5.4 shows the results of different deep learning models, FARASA, and MADAMIRA on the AQMAR dataset.

| Model | PER | LOC | ORG | Avg. F1 |
|---|-------|-------|-------|---------|
| FARASA | 0.681 | 0.431 | 0.373 | 0.495 |
| MADAMIRA | 0.401 | 0.262 | 0.228 | 0.297 |
| LSTM | 0.712 | 0.450 | 0.364 | 0.508 |
| Ensemble Avg. of 2 LSTMs | 0.728 | 0.437 | 0.374 | 0.513 |
| Deep Co-learning of 2 LSTMs using unlabeled data | 0.724 | 0.441 | 0.389 | 0.518 |
| Deep Co-learning of 2 LSTMs using semi-labeled data | 0.740 | 0.453 | 0.422 | 0.538 |

Table 5.4: Results of the models and Arabic NER tools on the AQMAR dataset

5.2.3 NEWS Dataset

After we had found the best models, we tested our models on the NEWS dataset. NEWS dataset is an annotated corpus for the task of NER. The corpus was de-

veloped by Darwish [36]. The NEWS dataset contains 292 sentences that were retrieved from the RSS feed of the Arabic (Egypt) version of news.google.com from Oct. 6, 2012. The corpus contains news from different sources and covers international and local news, politics, financial news, health, sports, entertainment, and technology. Table 5.5 shows the results of different deep learning models, FARASA, and MADAMIRA on the NEWS dataset.

| Model | PER | LOC | ORG | Avg. F1 |
|---|------------|------------|------------|----------------|
| FARASA | 0.695 | 0.731 | 0.421 | 0.639 |
| MADAMIRA | 0.525 | 0.289 | 0.208 | 0.340 |
| LSTM | 0.845 | 0.650 | 0.464 | 0.653 |
| Ensemble Avg. of 2 LSTMs | 0.847 | 0.662 | 0.458 | 0.655 |
| Deep Co-learning of 2 LSTMs using unlabeled data | 0.855 | 0.623 | 0.488 | 0.655 |
| Deep Co-learning of 2 LSTMs using semi-labeled data | 0.848 | 0.662 | 0.494 | 0.668 |

Table 5.5: Results of the models and Arabic NER tools on the NEWS dataset

5.2.4 TWEETS Dataset

After we had found the best models, we tested our models on the TWEETS dataset. TWEETS dataset is an annotated corpus for the task of NER. The corpus was developed by Darwish [36]. The TWEETS dataset contains 982 tweets were randomly selected from tweets authored between November 23, 2011 and

November 27, 2011. He retrieved the tweets from Twitter by using the query “lang:ar“ (language=Arabic). Table 5.6 shows the results of different deep learning models, FARASA, and MADAMIRA on the TWEETS dataset.

| Model | PER | LOC | ORG | Avg. F1 |
|---|------------|------------|------------|----------------|
| FARASA | 0.398 | 0.475 | 0.957 | 0.399 |
| MADAMIRA | 0.328 | 0.212 | 0.135 | 0.225 |
| LSTM | 0.592 | 0.318 | 0.285 | 0.398 |
| Ensemble Avg. of 2 LSTMs | 0.610 | 0.325 | 0.293 | 0.409 |
| Deep Co-learning of 2 LSTMs using unlabeled data | 0.634 | 0.324 | 0.340 | 0.432 |
| Deep Co-learning of 2 LSTMs using semi-labeled data | 0.652 | 0.333 | 0.353 | 0.446 |

Table 5.6: Results of the models and Arabic NER tools on the TWEETS dataset

5.2.5 Discussion

In Table 5.4, we notice that using Deep Co-learning LSTM models perform better than FARASA, MADAMIRA, the supervised LSTM, and the supervised ensemble averaging of 2 LSTMs on the AQMAR dataset. The Arabic NER tools FARASA and MADAMIRA have an F-measure of 0.495 and 0.297, respectively. The supervised model LSTM and the ensemble averaging of 2 LSTMs that were trained only on the training data of ANERCorp have an F-measure of 0.508 and 0.513, respectively. Comparing the two Deep Co-learning models we obtain a

higher F-measure of 0.538 using semi-labeled data.

In Table 5.5, we remark that FARASA has a high F-measure of 0.639 comparing to MADAMIRA on the NEWS dataset but a lower F-measure comparing to the models. MADAMIRA has the lowest F-measure of 0.340. The supervised models, LSTM and ensemble averaging of 2 LSTMs, perform better than MADAMIRA with an F-measure of 0.653 and 0.655, respectively. Our Deep Co-learning of 2 LSTMs using unlabeled data performs better than the supervised LSTM model but equally with the supervised ensemble averaging of 2 LSTMs. The model that obtains the highest F-measure is the Deep Co-learning of 2 LSTMs using semi-labeled data with an F-measure of 0.668.

In Table 5.6, also we remark that FARASA has a high F-measure of 0.399 comparing to MADAMIRA on the TWEETS dataset but a lower F-measure comparing to the models. Also for the TWEETS dataset, MADAMIRA has the lowest F-measure of 0.225. The supervised models, LSTM and ensemble averaging of 2 LSTMs, perform better than MADAMIRA with an F-measure of 0.398 and 0.409, respectively. Our Deep Co-learning models perform better than the supervised models. Comparing the two Deep Co-learning models, we obtain a higher F-measure using semi-labeled data of 0.446. In this dataset, we are getting lower F-measures than any dataset because tweets are subject to mistakes and misspellings.

We conclude that our Deep Co-learning of 2 LSTMs using semi-labeled data is making a significant improvement comparing to the Arabic NER tools, supervised

models, and the Deep Co-learning of 2 LSTMs using unlabeled data. As we previously said, the Deep Co-learning models were not run for an extended period, so we expect that if we run the semi-supervised models for much more than 50 iterations, we will get better results.

5.2.6 Error Analysis

In this part, we are going to analyze the error output of the Deep Co-learning models. We noticed that the Deep Co-learning that uses semi-labeled data and the one that uses unlabeled data tag universities as an ORG instead of a LOC according to the AQMAR dataset. For examples **جامعة براون** and **ماستشوستش معهد**. Also, the two versions of Deep Co-learning are tagging places of worship and some historical places as OTHER label instead of a location label. For examples, **المعهد الروماني جويتر** and **الجامع الاموي**. As for the PER label, our Deep co-learnings are making mistakes when it comes to named entities that represent people of an empire or people from a continent. The two version of Deep Co-learning is predicting those named entities as OTHER like **للاوربيين** and **العثمانيين**. Even between the two versions of Deep Co-learning, they are some predictions that are wrong for example the Deep Co-learning that uses unlabeled data predicted **ارامكو** as an OTHER label instead of an ORG label, while the Deep Co-learning that uses semi-labeled data predicted it correctly. Another example, in this tweet **البنك العقاري يعطيك قرض ٥٠٠ الف ريال** the Deep

Co-learning that uses semi-labeled data tagged ريال correctly while the one that uses unlabeled data predicted as ORG, which is wrong. The ORG label might be explained that the Deep Co-learning that uses unlabeled data has been only trained with data that contains the named entity ريال مدريد . According to the evaluations the Deep Co-learning that uses semi-labeled data performed better than any Arabic NER tools and deep learning models, but our model still needs improvements.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this work, we presented a new approach to detect and classify NEs in the Arabic language. Our approach is based on a semi-supervised algorithm called co-training that we adapted in the context of deep learning, which we named it Deep Co-learning. Deep Co-learning can be trained on labeled and semi-labeled data.

We developed a Wikipedia Named Entity Recognizer, a deep learning model, to predict the named entity of a giving article by only analyzing the summary part. Then we generated a large semi-annotated corpus from Wikipedia by predicting the labels of the named entities that have an internal link using the Wikipedia Named Entity Recognizer.

Finally, we trained our Deep Co-learning models using labeled and semi-

labeled data. Our final model outperformed standard Arabic NER tools (FARASA and MADAMIRA) and fully-supervised model on different datasets.

6.2 Future Work

As for the future work we are planning to train the Deep Co-learning models for a longer period so that we can augment the labeled dataset with more instances. Also, We aim to test our model on other NLP tasks like Part-of-speech tagging. Besides, We seek to improve our approach by adding features like morphological features.

Appendix A

Abbreviations

| | |
|------|--|
| SRL | Semantic Role Labeling |
| NE | Named Entity |
| NER | Named Entity Recognition |
| LSTM | Long Short-Term Memory |
| NLP | Natural Language Processing |
| RNN | Recurrent Neural Networks |
| CBOW | Continuous Bag of Words |
| BAMA | Buckwalter's Arabic Morphological Analyzer |
| MSA | Modern Standard Arabic |
| ML | Machine Learning |
| CRF | Conditional Random Field |
| POS | Part Of Speech |

BPC Base Phase Chunking
SVM Support Vector Machine

Bibliography

- [1] N. Y. Habash, “Introduction to arabic natural language processing,” *Synthesis Lectures on Human Language Technologies*, vol. 3, no. 1, pp. 1–187, 2010.
- [2] K. Shaalan, “A survey of arabic named entity recognition and classification,” *Computational Linguistics*, vol. 40, no. 2, pp. 469–510, 2014.
- [3] I. Goodfellow, Y. Bengio, and A. Courville, “Deep learning.” Book in preparation for MIT Press, 2016.
- [4] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [5] T. Buckwalter, “Buckwalter {Arabic} morphological analyzer version 1.0,” 2002.
- [6] J. Halpern *et al.*, “Lexicon-driven approach to the recognition of arabic named entities,” in *Second International Conference on Arabic Language Resources and Tools*, 2009.
- [7] I. Alkharashi, “Person named entity generation and recognition for arabic language,” in *Proceedings of the Second International Conference on Arabic Language Resources and Tools*, pp. 205–208, Citeseer, 2009.
- [8] Y. Benajiba, P. Rosso, and J. M. Benedíruiz, “Anersys: An arabic named entity recognition system based on maximum entropy,” in *International Conference on Intelligent Text Processing and Computational Linguistics*, pp. 143–153, Springer, 2007.
- [9] Y. Benajiba, M. Diab, and P. Rosso, “Arabic named entity recognition: A feature-driven study,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 5, pp. 926–934, 2009.
- [10] K. Shaalan and H. Raza, “Nera: Named entity recognition for arabic,” *Journal of the American Society for Information Science and Technology*, vol. 60, no. 8, pp. 1652–1663, 2009.

- [11] Y. Benajiba, M. Diab, and P. Rosso, “Arabic named entity recognition using optimized feature sets,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 284–293, Association for Computational Linguistics, 2008.
- [12] J. Maloney and M. Niv, “Tagarab: a fast, accurate arabic name recognizer using high-precision morphological analysis,” in *Proceedings of the Workshop on Computational Approaches to Semitic Languages*, pp. 8–15, Association for Computational Linguistics, 1998.
- [13] S. Abuleil, “Extracting names from arabic text for question-answering systems,” in *Coupling approaches, coupling media and coupling languages for information retrieval*, pp. 638–647, LE CENTRE DE HAUTES ETUDES INTERNATIONALES D’INFORMATIQUE DOCUMENTAIRE, 2004.
- [14] D. Samy, A. Moreno, and J. M. Guirao, “A proposal for an arabic named entity tagger leveraging a parallel corpus,” in *International Conference RANLP, Borovets, Bulgaria*, pp. 459–465, 2005.
- [15] S. Mesfar, “Named entity recognition for arabic using syntactic grammars,” in *Natural Language Processing and Information Systems*, pp. 305–316, Springer, 2007.
- [16] K. Shaalan and H. Raza, “Person name entity recognition for arabic,” in *Proceedings of the 2007 Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources*, pp. 17–24, Association for Computational Linguistics, 2007.
- [17] G. R. Doddington, A. Mitchell, M. A. Przybocki, L. A. Ramshaw, S. Strassel, and R. M. Weischedel, “The automatic content extraction (ace) program-tasks, data, and evaluation.,” in *LREC*, vol. 2, p. 1, 2004.
- [18] A. Elsebai, F. Meziane, and F. Z. Belkredim, “A rule based persons names arabic extraction system,” *Communications of the IBIMA*, vol. 11, no. 6, pp. 53–59, 2009.
- [19] R. Al-Shalabi, G. Kanaan, B. Al-Sarayreh, K. Khanfar, A. Al-Ghonmein, H. Talhouni, and S. Al-Azazmeh, “Proper noun extracting algorithm for arabic language,” in *International conference on IT, Thailand*, 2009.
- [20] Y. Benajiba and P. Rosso, “Anersys 2.0: Conquering the ner task for the arabic language by combining the maximum entropy with pos-tag information.,” in *IICAI*, pp. 1814–1823, 2007.
- [21] Y. Benajiba, M. Diab, P. Rosso, *et al.*, “Arabic named entity recognition: An svm-based approach,” in *Proceedings of 2008 Arab International Conference on Information Technology (ACIT)*, pp. 16–18, 2008.

- [22] Y. Benajiba, I. Zitouni, M. Diab, and P. Rosso, “Arabic named entity recognition: using features extracted from noisy data,” in *Proceedings of the ACL 2010 conference short papers*, pp. 281–285, Association for Computational Linguistics, 2010.
- [23] A. Abdul-Hamid and K. Darwish, “Simplified feature set for arabic named entity recognition,” in *Proceedings of the 2010 Named Entities Workshop*, pp. 110–115, Association for Computational Linguistics, 2010.
- [24] “Anercorp. <http://www1.ccls.columbia.edu/ybenajiba/downloads.html>.”.
- [25] S. AbdelRahman, M. Elarnaoty, M. Magdy, and A. Fahmy, “Integrated machine learning techniques for arabic named entity recognition,” *IJCSI*, vol. 7, pp. 27–36, 2010.
- [26] R. Koulali and A. Meziane, “A contribution to arabic named entity recognition,” in *ICT and Knowledge Engineering (ICT & Knowledge Engineering), 2012 10th International Conference on*, pp. 46–52, IEEE, 2012.
- [27] A. Pasha, M. Al-Badrashiny, M. T. Diab, A. El Kholy, R. Eskander, N. Habash, M. Pooleery, O. Rambow, and R. Roth, “Madamira: A fast, comprehensive tool for morphological analysis and disambiguation of arabic,” in *LREC*, vol. 14, pp. 1094–1101, 2014.
- [28] A. Abdelali, K. Darwish, N. Durrani, and H. Mubarak, “Farasa: A fast and furious segmenter for arabic,” in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pp. 11–16, Association for Computational Linguistics, San Diego, California, 2016.
- [29] S. Abdallah, K. Shaalan, and M. Shoaib, “Integrating rule-based system with classification for arabic named entity recognition,” in *International Conference on Intelligent Text Processing and Computational Linguistics*, pp. 311–322, Springer, 2012.
- [30] M. Oudah and K. F. Shaalan, “A pipeline arabic named entity recognition using a hybrid approach,” in *COLING*, pp. 2159–2176, 2012.
- [31] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.
- [32] F. A. Gers and J. Schmidhuber, “Recurrent nets that time and count,” in *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*, vol. 3, pp. 189–194, IEEE, 2000.
- [33] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, “How to construct deep recurrent neural networks,” *arXiv preprint arXiv:1312.6026*, 2013.

- [34] A. Blum and T. Mitchell, “Combining labeled and unlabeled data with co-training,” in *Proceedings of the eleventh annual conference on Computational learning theory*, pp. 92–100, ACM, 1998.
- [35] B. Mohit, N. Schneider, R. Bhowmick, K. Oflazer, and N. A. Smith, “Recall-oriented learning of named entities in arabic wikipedia,” in *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 162–173, Association for Computational Linguistics, 2012.
- [36] K. Darwish, “Named entity recognition using cross-lingual resources: Arabic as an example.,” in *ACL (1)*, pp. 1558–1567, 2013.
- [37] M. A. Zahran, A. Magooda, A. Y. Mahgoub, H. Raafat, M. Rashwan, and A. Atyia, “Word representations in vector space and their applications for arabic,” in *International Conference on Intelligent Text Processing and Computational Linguistics*, pp. 430–443, Springer, 2015.
- [38] J. Pennington, R. Socher, and C. Manning, “Glove: Global Vectors for Word Representation,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 1532–1543, Association for Computational Linguistics, Oct. 2014.
- [39] R. Al-Rfou, B. Perozzi, and S. Skiena, “Polyglot: Distributed word representations for multilingual nlp,” *arXiv preprint arXiv:1307.1662*, 2013.
- [40] B. Gao, J. Bian, and T.-Y. Liu, “Wordrep: A benchmark for research on learning word representations,” *arXiv preprint arXiv:1407.1640*, 2014.