A Semi-supervised Learning Approach to Arabic Named Entity Recognition

Maha Althobaiti, Udo Kruschwitz, and Massimo Poesio
School of Computer Science and Electronic Engineering
University of Essex
Colchester, UK
{mjaltha, udo, poesio}@essex.ac.uk

Abstract

We present ASemiNER, a semi-supervised algorithm for identifying Named Entities (NEs) in Arabic text. ASemiNER does not require annotated training data, or gazetteers. It also can be easily adapted to handle more than the three standard NE types (Person, Location, and Organisation). To our knowledge, our algorithm is the first study that intensively investigates the semi-supervised pattern-based learning approach to Arabic Named Entity Recognition (NER). We describe ASemiNER and compare its performance with different supervised systems. We evaluate this algorithm by way of experiments to extract the three standard named-entity types. Ultimately, our algorithm outperforms simple supervised systems and also performs well when we evaluate its performance in order to extract three new, specialised types of NEs (Politicians, Sportspersons, and Artists).

1 Introduction

Named Entities (NEs) are textual references via proper names, such as first and last names, locations, and companies. Detecting NEs within unstructured text and classifying them into predefined categories of names is known as Named Entity Recognition (NER) (Grishman and Sundheim, 1996).

Arabic NER has been given great amount of attention over the past fifteen years. A number of Arabic NER systems have been developed using three approaches, which have been investigated thoroughly in the literature of NER. These approaches are rule-based (Shaalan and Raza, 2007; Shaalan and Raza, 2009), Machine Learning (ML) (Benajiba et al., 2007; Benajiba and Rosso, 2007; Benajiba and Rosso, 2008; Abdul-Hamid and Darwish, 2010) and hybrid (Abdallah et al., 2012; Oudah and Shaalan, 2012).

Over the past decade, some studies have explored the possibility of solving the problem of NER with a reduced level of supervision. These studies proposed semi-supervised and unsupervised systems, which no longer require annotated datasets and can be easily adapted to new types (Nadeau et al., 2006; Etzioni et al., 2005; Liao and Veeramachaneni, 2009; Liu et al., 2011).

This paper introduces ASemiNER, an Arabic semi-supervised NER system built under minimal supervision. Gazetteers (predefined lists of NEs) and annotated corpora are not required by ASemiNER. That is, ASemiNER is a bootstrapping algorithm that takes a few examples of a particular NE type as input and iteratively induces and learns patterns, which are used to extract more examples. Extraction patterns are induced and generalised automatically from data using very general criteria that require no human intervention, and no prior knowledge of the language or the corpus domain. In addition to the fact that ASemiNER extracts and recognises the three standard NEs (Person, Location, and Organisation names), it has proven to be an adaptable system that can be easily modified to extract new NEs without the need for analysing the dataset or collecting and tagging new large corpora.

The remainder of this paper is structured as follows: Section 2 includes background information on Arabic NER, including recent work. Section 3 illustrates the architecture of the proposed algorithm. Section 4 describes the corpora used in the experiments and the preprocessing steps used to prepare them. The experimental setup and the evaluation results are reported and discussed in Section 5. Finally, the conclusion features comments regarding our future work.

Proceedings of Recent Advances in Natural Language Processing, pages 32–40,
Hissar, Bulgaria, 7-13 September 2013.

2 Background

2.1 State-of-the-art Arabic NER

Arabic has started to gain a significant amount of focus in large-scale projects, such as Global Autonomous Language Exploitation (GALE)[1] (Nadeau and Sekine, 2007). In addition, researchers have been making an effort over the past fifteen years to boost the performance of Arabic NER task.

Many Arabic NER researchers have employed rule-based techniques (Mesfar, 2007; Shaalan and Raza, 2009) that require experts. Thus, many ML methods, including Supervised Learning (SL) techniques, have been investigated in order to learn NE annotated decisions from training data. The most common SL techniques used for NER are Maximum Entropy (Benajiba et al., 2007), Support Vector Machine (Benajiba et al., 2008), and Conditional Random Fields (CRF) (Benajiba and Rosso, 2008)

Abdallah et al. (2012) proposed a hybrid NER system for Arabic in which they integrate the rule-based approach with the ML-based approach in order to optimise overall performance. Oudah and Shaalan (2012) contribute to the Arabic hybrid NER approach by investigating three different ML approaches including Decision Trees, SVM, and Logistic Regression, along with different features. Their system outperforms the state-of-the-art Arabic NER when applied to ANERcorp.

AbdelRahman et al. (2010) presented an integration approach between two machine learning techniques, CRF and semi-supervised pattern generation where the generated patterns were used as CRF features. Mohit et al. (2012) also investigated the problem of NER in Arabic Wikipedia using semi-supervised domain adaptation technique. They trained a model on newswire text based on standard supervised method. Then, they adapted the model with self-training on unlabeled target-domain data.

2.2 Semi-supervised techniques

Semi-supervised learning (SSL) is a relatively recent approach in the NLP community. It is still active and is likely to be improved and tested with various NLP tasks, including NER. The most common SSL technique is bootstrapping, which only requires minimal supervision, namely, a set of seeds in order to initiate the learning process (Nadeau and Sekine, 2007).

An early study that influenced later works (Riloff and Jones, 1999) propounds that the algorithm begins with a set of seed examples of a particular entity type (e.g., London is entity of type city). Then, all contexts (e.g., "State of <X >", "seminars in <X >") found around these seeds in a large corpus will be gathered, ranked, and used to find new examples. Pasca et al. (2006) used the same bootstrapping technique employed in (Riloff and Jones, 1999), but they applied the technique to very large corpora and managed to generate one million facts with a precision rate of about 88%.

Etzioni et al. (2005) proposed a system called "KnowItAll" that aims to automate the process of extracting large collections of facts, such as names of cities or movies from the web, in a domain-independent and scalable manner, starting with a set of predicates (e.g., City, and Country) and a set of generic extraction patterns. Furthermore, Nadeau et al. (2006) proposed a named-entity recognition system that combines named entity extraction inspired by the study of Etzioni et al. (2005) with a simple form of named-entity disambiguation. Their study's remarkable performances compete with baseline supervised approaches.

In 2009, Liao and Veeramachaneni proposed a simple semi-supervised learning algorithm using CRF. the algorithm starts with a small amount of labeled data *(*L) and a classifier that is trained on *L*. Then, the data *D* are extracted from unlabeled data using the trained classifier. The extracted data *D* with high confidence are added to the training data. At each iteration, the classifier trained on the previous training data is used to tag unlabeled data and so on (Liao and Veeramachaneni, 2009). Baroni et al. (2010) presented an algorithm that induces semantic information from naturally occurring text without supervision and requiring a small amount of pre-encoded knowledge, POS tagging, lemmatization of the corpus, and a set of extraction templates defined over POS sequences.

3 Methodology

Like most other semi-supervised algorithms, our algorithm contains 3 components, as shown in Figure 1.

[1] http://projects.ldc.upenn.edu/gale/

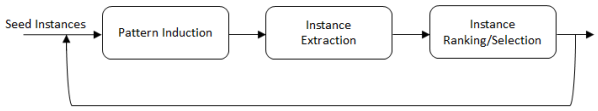

Figure 1: The Three Components of ASemiNER

Our algorithm begins with a seed list of a few examples of a given NE type (e.g., 'Muhammad' and 'Obama' can be used as seed instances for entity of type person) and learns patterns that are used to extract more examples (candidate NEs). These examples will be sorted and used again as seed instances for the next iteration.

3.1 Pattern Induction

3.1.1 Initial Patterns

ASemiNER uses a similar approach to that which was adopted in Baroni et al. (2010) to infer patterns, but with some modifications. Our algorithm infers a set of surface patterns that contain seed instances in the training corpus. So, for each seed instance x, we first retrieve all sentences containing the term x. Since words preceding or following the target word may be useful for determining its category, the algorithm extracts a number of tokens[2] on each side of the seed x without crossing sentence boundaries. Figure 2 is an example of initial patterns containing the seed instance (Muhammad) and its surrounding tokens.

Arabic Pattern:

- نوه / VBD الدكتور/NN محمد/NNP البشر/NNP سفير/NN المملكة/NN العربية/JJ السعودية/JJ في/IN المغرب/NNP

English Gloss:

- Dr. /NN Mohammed/NNP Albshr/NNP the/DT ambassador/NN of/IN Saudi/NNP Arabia/NNP in/IN Morocco/NNP indicated/VBD that/DT

Figure 2: Example of Initial Pattern

We will refer to each "Token/POS-tag" pair as "TP pair" (e.g., 'indicated/VBD' represents one TP pair). Noun tokens in TP pairs are kept in their inflected form, while verb tokens are replaced with their roots. For example, (*katabt* 'wrote')[3] and (*taktub* 'writes') will be changed to (*katab* 'write').

For each particular type of NEs (e.g., Person), lists of "trigger" words[4] (nouns and verbs) are provided as input. The lists of trigger nouns are semi-automatically extracted from randomly selected Arabic Wikipedia articles. Specifically, we extract nouns that appear most frequently before or after the NE and stored them as trigger nouns. Trigger verbs are the most frequent verbs (stems) that appear before or after NE in the Arabic Wikipedia articles. Trigger verbs and nouns, which surround NEs, are identified in order to find the most common Arabic NE indicators. Some examples of trigger nouns are: (*alsayd* 'Mr.'), (*alsaydh* 'Mrs.'), and (*bn* 'the son of') for a person's name; (*madynah* 'city'), and (*wilaayah* 'state') for location.

3.1.2 Generalisation

In the next step, the initial patterns are generalised. Therefore, all extracted initial patterns should complete the following steps in order to generate the final patterns:

1. TP pairs that contain nouns, and verbs are stripped of their "Token" parts, unless they are in the corresponding lists of trigger words. For example, TP pair (*alsayd/NN* 'Mr./NN') will stay unchanged since (*alsayd* 'Mr.') is in the list of trigger nouns, while (*qalam/NN* 'pen/NN') will be changed to only ' */ NN*' as (*qalam* 'pen') is not among trigger nouns.

2. TP pairs that contain prepositions are not changed.

3. TP pairs that contain other parts of speech categories (e.g., proper noun, adjective, coordinating conjunction) are stripped of their "Token" parts. For instance, the token (*mufyd/JJ* 'useful/JJ') will be converted to only ' */JJ*' without the "Token" part.

4. All POS tags used for verbs (e.g., *VBP, VBD, VBN*) are converted to one form: *VB*.

5. All POS tags used for nouns (e.g., *NN, NNS*) are converted to one form: *NN*.

6. All POS tags used for proper nouns (e.g., *NNP, NNPS*) are converted to one form: *NNP*.

[2] Following a few trials, we found that a suitable number of tokens is 7.

[3] Throughout the entire paper, Arabic words are represented as follows: (*Qalam transliteration* 'English translation').

[4] Also known as keywords or indicators that form a window around the NE.

7. The seed instance is replaced with NE class tag (e.g., <PersonName>, <Location>, <Organisation>).

Figure 3 shows the final pattern resulting from the initial pattern, after the constrained processes mentioned above are applied:

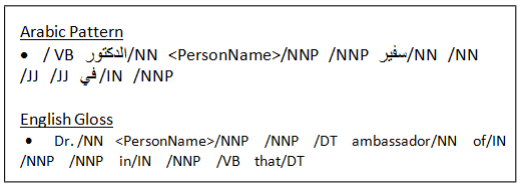

Figure 3: Example of Final Pattern Produced by ASemiNER

All final patterns that are generated from the algorithm and their frequencies are first computed, and then gathered to form the pattern set (P). In the final step, two more patterns were generated from every pattern in P. Therefore, the algorithm split every final pattern into two parts, where each seed instance is located in the leftmost or rightmost position in the pattern. The two patterns generated from our previously mentioned example can be seen in Figure 4.

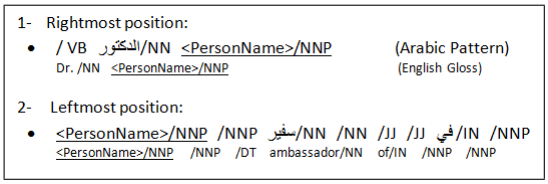

Figure 4: Two More Patterns Generated from the Final Pattern

The rationale behind this is to increase the generality of the patterns by making them shorter in length, thus increasing their ability to collect more candidate NEs in the matching process against the text. For example, the short pattern "*Dr./NN <PersonName >*" might successfully match more NEs in the text than the long pattern illustrated in Figure 3. However, short patterns, which have TP-pairs containing no "Token" parts at all, but POS-taggings, are a source of noise. Therefore, the final patterns set (P) is filtered every time a new pattern is added to it. Thus, repeated patterns are not added. In addition, any pattern consisting of less than 6 TP-pairs[5] should contain at least one TP-pair with "Token" part. Consequently, the pattern "*/VB /NN <PersonName >/NNP /NNP*" is rejected and not added to the set (P).

[5]Informal experiments show us that a pattern with less than six TP-pairs is more likely to be a noisy pattern, especially if its TP-pairs do not contain "Token" parts at all.

3.2 Instance Extraction

In this phase, ASemiNER retrieves the set of instances I from the training corpus that match any of the patterns in P. First of all, we should make sure that the generalisation steps used in inducing patterns are applied to the training corpus in order to prepare it for the matching process (e.g., VBD, VBP, and VBN are converted to VB and so on). The matching final patterns in P against the corpus is conducted using regular expressions (regex). For example, the regex for the pattern " */VB alductur/NN <PersonName >/NNP*" is depicted in Figure 5.

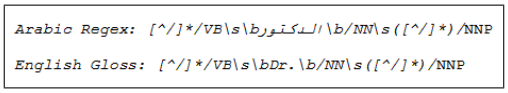

Figure 5: Regex Automatically Generated from a Final Pattern

Since the absence of capitalisation in Arabic, Arabic POS taggers might mistake some organisations and locations for nouns (*NN*) or adjectives (*JJ*), especially meaningful names. For example, (*alwlaayaat almutHdah alamrykyh* 'United States of America') might be tagged as *alwlaayaat/NNS almutHdah/JJ alamrykyh/JJ*. The ASemiNER system automatically generates regexes from final patterns without modifying them, regardless of whether the POS tags assigned to the proper nouns by POS tagger are accurate or not.

An informal experiment showed that most proper Arabic names are 2 or 3 tokens in length. Therefore, in order to increase the number of NEs collected in each iteration, we allowed the ASemiNER system to automatically add the information of average NE length to the produced regexes, as seen below:

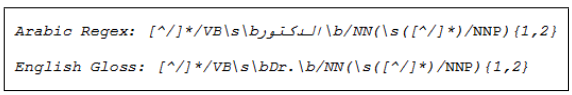

Figure 6: Regex with Average NE Length

We have also noticed that increasing the average length of proper names to more than 2 tokens increases the recall but negatively affects the precision and quality of the collected NEs.

3.3 Instance Ranking/Selection

ASemiNER ranks all examples[6] in *I* according to the number of different patterns that are used to extract them (Baroni et al., 2010). For example, candidate NE that is extracted by 5 distinct patterns will be ranked before the one that is extracted by only 2 distinct patterns. We avoid the use of plain frequencies as a criterion since some bad examples appear more in the text in a relatively similar context and can be extracted by only one pattern in (*P*). Meanwhile, the good examples might appear less in the text, but in different contexts, and can be extracted by more than one pattern in (*P*). Therefore, the high frequency threshold does not always produce good examples. In addition, pattern variety is a better cue to semantics than absolute frequency.

ASemiNER ranks the examples according to distinct patterns, and discards all but the top *m*, where *m* is set to the number of examples from the previous iteration, plus one. These *m* instances will be used in the next iteration, and so on. For example, if we start the algorithm with 10 seed instances, the following iteration will start with 11, and the next one will start with 12, and so on. This procedure is necessary in order to ensure that bad instances from the previous iteration are not included in the next one.

Moreover, information theory approaches is commonly used in text mining (Turney et al., 2010). For that reason, we tried to apply an Information-theory approach to examine the plausibility of candidate NEs, which are extracted by our system. Hence, we used Pointwise Mutual Information (PMI) statistics to measure the association strength of the instance i in (*I*) across each pattern in (*P*). A reliable instance is one that is associated with as many patterns in *P* as possible.

$$pmi(i) = \sum_{p \epsilon P} log \frac{|i,p|}{|i|*|p|}$$

In this case, $|i,p|$ is the frequency of the instance *i* extracted by pattern *p*. $|i|$ is the frequency of the instance in the corpus. The corpus should be decliticized, clitics should be separated from words, in order to reduce data sparseness and to compute the correct frequencies for each word in the corpus text sequence. $|p|$ is the frequency of the pattern *p* in the corpus.

[6] Also known as instances or candidate NEs.

4 Datasets

ASemiNER does not require any kind of annotated corpora or any type of gazetteers. However, our selection of corpora was based on the intention to compare ASemiNER with other systems. We chose two commonly used corpora in order to evaluate and compare our system with existing systems. These datasets are ANERcorp and ACE 2005.

ANERcorp contains more than 150,000 tokens (11% of the tokens are NEs). It is composed of a training corpus and a test corpus built and tagged especially for the NER task by Benajiba et al. (2007). We chose to evaluate our proposed system with the ANERcorp test corpus because it is commonly used in literature for comparing with existing systems. More details about ANERcorp are given in (Benajiba et al., 2007).

The second dataset used in the training phase is ACE 2005. It is available from the Linguistic Data Consortium (LDC) and has more than 113,000 tokens. The genres utilised in ACE 2005 are Broadcast News, NewsWire, and WebLogs.

Ten percent of the training data was dedicated to the validation set which was used to validate the effectiveness of the trained models. It also helped assign appropriate values to several parameters in our system, such as the number of initial seeds, the criterion to stop the training process, and so on.

ANERcorp and ACE corpora were preprocessed in order to prepare them for our proposed algorithm. Thus, sentence detection was applied to the corpora. Then, we conducted clitic tokenization, since neglecting clitics may cause a loss of important information when generating the patterns. We chose decliticization scheme 'D2' in which conjunctions, prepositions, and future marks are separated from each token (Habash and Sadat, 2006).

Each verb in the corpus is changed to its root from which it is derived. We used root stemmer, namely Khoja's stemmer (Khoja and Garside, 1999), instead of using a light stemmer, which sometimes fails to conflate related forms that should group together, as our goal was to produce a sound set of general patterns.

Regarding POS-tagging, we used AMIRA toolkit (Diab, 2009) and chose Reduced Tag Set (RTS), which neglects inflections in Arabic word categories, since our proposed method does not require any deep morphological information related

to gender, number, or definiteness. This information is unnecessary, considering our aim is to make the algorithm generally applicable to languages other than Arabic.

5 Experiments & Results

We developed several experimental models according to three parameters that are defined in our proposed algorithm: the number of initial seeds, the ranking measure, and the number of iterations. The ANERcorp test corpus was used to evaluate every trained model. Regarding the NE type, we had two levels of experiments: in Experiment 1 we trained models to identify the standard NEs (Person, Location, Organisation) in order to compare our system with existing systems; Experiment 2 involved the identification of specialised NEs (Politicians, Sportspersons, and Artists).

5.1 Experiment 1: Standard NEs

We started with a simple model, which was trained on the ANERcorp corpus and passed through the three components only once. For each NE class, we only started with five seed instances. We referred to this model as 'Simple-Model-5'. We also trained two more models, Simple-Model-10 and Simple-Model-20, which only differed from Simple-Model-5 in the number of seed instances for each NE class; the number of seeds were 10 and 20 respectively.

Table 1 shows the precision and recall of these models for each NE class when applying them to the ANERcorp test corpus.

		Simple-Model-5	Simple-Model-10	Simple-Model-20
Person	Precision	88.09	88.46	86.32
	Recall	36.01	39.06	43.20
	F-measure	51.12	54.19	57.58
Location	Precision	89.96	86.80	90.55
	Recall	51.88	55.10	55.45
	F-measure	65.81	67.41	68.78
Organisation	Precision	85.95	83.87	84.35
	Recall	22.38	23.66	27.67
	F-measure	35.51	36.91	41.67

Table 1: Results of Simple-Model-5, Simple-Model-10, and Simple-Model-20

Based on these results, the number of iterations was set to ten for all coming experiments, because we recognised that increasing the number of iterations to more than ten loops makes no significant improvement in the performance of the system (improvement <0.01). We started with 20 seed instances for each NE class and the training corpus was ANERcorp. Candidate NEs were ranked according to the number of distinct patterns in order to select those that ranked the highest as seeds for the next iteration, as explained in section 3.3. We referred to these trained models, one model for each NE class, as 'Model-A(NE class)'.

We also used Pointwise Mutual Information (PMI) as a ranking measure for candidate NEs instead of using the number of distinct patterns. Table 2 shows the outcome of evaluating the trained models on the ANERcorp test corpus.

The results obtained using PMI as a measure to select the seed instances for the next iteration revealed generally low performance and particularly low recall. This can be attributed to the PMI's biased towards infrequent words (Turney et al., 2010), which means less patterns are extracted for the next iteration. Using PMI, the precision was not affected at all, since very few patterns are added into set *P* in each iteration. In general, PMI results in a performance lower than that achieved when using the number of distinct patterns as a reliable measure for seed selection.

		Model-A	Model-A (PMI)
Person	Precision	85.91	85.97
	Recall	51.17	44.64
	F-measure	64.14	58.77
Location	Precision	87.96	90.62
	Recall	62.48	56
	F-measure	73.06	69.22
Organisation	Precision	84.27	85.54
	Recall	40.30	33.80
	F-measure	54.52	48.45

Table 2: The Performance of Model-A on the ANERcorp Test Corpus and the Effect of Using PMI

In the next step, a large corpus, which is a combination of the ANERcorp training set and ACE 2005, was used in the training phase. We referred to the trained models resulting from this experiment as 'Model-B(NE class)'. Using large training data increases the recall of the trained models with a small negative effect on precision. However, the total F-measure is better when using a large corpus, rather than training our model on small training data.

Table 3 summarises the trained models with their values for each parameter. It also shows the performance of each model when applying them to

the ANERcorp test corpus by computing its average F-measure for the three standard NEs: person, location, and organisation.

Trained Models	Training Corpus	Ranking measure	No. of initial seeds	Avg. F-measure
Simple Model-5	ANERcorp	--	5	50.81
Simple Model-10	ANERcorp	--	10	52.83
Simple Model-20	ANERcorp	--	20	56.01
Model-A	ANERcorp	Number of distinct Patterns	20	63.91
Model-A (PMI)	ANERcorp	Pointwise Mutual Information (PMI)	20	58.81
Model-B	ANERcorp + ACE 2005	Number of distinct Patterns	20	64.26

Table 3: Different Trained Models with their Parameters and their Performance on the ANERcorp Test Corpus

Based on all of our previous experiments, we have concluded that the following parameters give the best results: the number of initial seeds is 20, the number of iterations is 10, and the ranking measure is the number of distinct patterns used in extraction candidate NEs. Therefore, for the sake of simplicity, we refer to our system that used the trained models with the previously mentioned parameters as "ASemiNER".

In comparison with different supervised NER systems (Benajiba et al., 2007; Benajiba and Rosso, 2007; Benajiba and Rosso, 2008) when applied on the ANERcorp test corpus, ASemiNER can outperform a sensible supervised system, which depends on maximum entropy and a set of features. It still cannot compete, however, with more complex supervised systems. Table 4 shows the results of the comparison.

	Person	Location	Organisation
	F-measure	F-measure	F-measure
ANERsys 1.0	46.69	80.25	36.79
ANERsys 2.0	52.13	86.71	46.43
CRF-based System	73.35	89.74	65.76
Our System(ASemiNER)	64.14	73.06	54.52

Table 4: The Comparison Between Three Different Supervised Systems and our System when Applied on the ANERcorp Test Corpus

5.2 Experiment 2: Specialised NEs

Although most common types of entities investigated in literature are names of people, organisations, and locations, there are many specialised domains that require new annotated corpora and systems to recognise their special NEs (Althobaiti et al., 2012). The recent increase in the number of social networks and specialised domains shows the need to obtain systems that can be easily adapted to identify different, new types of NEs, regardless of the domains.

In this section, we show how well the system recognises new types of NEs, politicians, sportspersons and artists. These new types have been chosen because they constitute the largest percentage of persons' names in ANERcorp. Thus, all annotated persons' names in ANERcorp must be re-annotated using one of four tages: POL, ART, SPORT, and Other. First of all, a guideline was formulated to distinguish the attributes of each class where each new type has been defined, described, and determined. After that, one of the authors re-annotated test corpus for evaluation purposes.

Unlike supervised learning, which may require additional examples in the training data for new categories of NE, our semi-supervised approach used the ANERcorp training data without any addition or modification. The methodology was applied without any major modifications. The modification is only related to generating new lists of trigger nouns and verbs for each type of new NEs (i.e., politicians, sportspersons, artists). They were generated in the same way explained in section 3.1. We manually checked each list to retain only verbs that have a high probability to indicate a specific type of NE. So, verbs like (*entakhab* 'elect'), and (*Swwat* 'vote') can be useful in the case of politician entities.

The performance in this task is comparable to that of standard named entities. Table 5 compares the performance of ASemiNER when extracting standard NEs and the three specialised NEs.

		Precision	Recall	F-measure
Specialised types of NE	Politicians	82.87	50.72	62.93
	Sportspersons	86.56	42.14	56.68
	Artists	82	47.14	59.86
Classic types of NE	Person	85.91	51.17	64.14
	Location	87.96	62.48	73.06
	Organisation	84.27	40.30	54.52

Table 5: The Performance of ASemiNER on the ANERcorp Test Corpus in order to Extract Both Standard & Specialised NEs

For sportspersons, the low recall is possibly due

to the impact of the lower number of varied contexts in which seeds occur. So, sportspersons constitute only 19% of all person names that exist in the training corpus, and they occur in a few contexts. Thus, the diversity of contexts in which seeds appear plays an important role in obtaining a trained model with good performance. The remaining recall errors can be attributed to the diversity of categories. Accordingly, sportspersons can be broken down into other categories, such as "football players", "golfers" and "wrestlers". In contrast, politician entity recognition has a higher recall than sportspersons. This can be attributed to two facts: 1) Politicians make up 44% of the people names in the training corpus, and 2) An efficient model results from using initial seeds like 'Bush' or 'Muhammad', since such examples occur frequently and in a variety of contexts in the training corpus. Overall, our semi-supervised system proved to be easily adaptable when extending the NE hierarchy. In addition, ASemiNER performs just as well when recognising the standard person category. Even more, our system highlighted the importance of the manner in which initial seeds are chosen in any semi-supervised approach.

6 Conclusion

All in all, we advance the the state-of-the art Arabic NER by avoiding the need for supervision, adopting a novel solution for the Arabic NER problem, and handling specialised NE types. Our solution is a semi-supervised approach in which our system (ASemiNER) produces semantic information from naturally occurring text with limited supervision. Each NE type, therefore, only requires a seed list made up of a few examples. Furthermore, in terms of experiments, ASemiNER outperforms sensible supervised systems. Admittedly our algorithm does not perform as well as complex supervised systems, however, its extremely limited dependence on supervision more than compensates for this point. Moreover, ASemiNER can be easily adapted to identify new types of NEs and does not generate problems typical of supervised methods that require annotated training data, and demand more effort and time to extract specialised types of NEs.

References

Sherief Abdallah, Khaled Shaalan, and Muhammad Shoaib. 2012. Integrating rule-based system with classification for arabic named entity recognition. In *Computational Linguistics and Intelligent Text Processing*, pages 311–322. Springer.

Samir AbdelRahman, Mohamed Elarnaoty, Marwa Magdy, and Aly Fahmy. 2010. Integrated machine learning techniques for arabic named entity recognition. *IJCSI*, 7:27–36.

Ahmed Abdul-Hamid and Kareem Darwish. 2010. Simplified feature set for arabic named entity recognition. In *Proceedings of the 2010 Named Entities Workshop*, pages 110–115. Association for Computational Linguistics.

Maha Althobaiti, Udo Kruschwitz, and Massimo Poesio. 2012. Identifying named entities on a university intranet. In *Computer Science and Electronic Engineering Conference (CEEC), 2012 4th*, pages 94–99. IEEE.

Marco Baroni, Brian Murphy, Eduard Barbu, and Massimo Poesio. 2010. Strudel: A corpus-based semantic model based on properties and types. *Cognitive Science*, 34(2):222–254.

Yassine Benajiba and Paolo Rosso. 2007. Anersys 2.0: Conquering the ner task for the arabic language by combining the maximum entropy with pos-tag information. In *IICAI*, pages 1814–1823.

Yassine Benajiba and Paolo Rosso. 2008. Arabic named entity recognition using conditional random fields. In *Proc. of Workshop on HLT & NLP within the Arabic World, LREC*, volume 8, pages 143–153.

Yassine Benajiba, Paolo Rosso, and José Miguel Benedíruiz. 2007. Anersys: An arabic named entity recognition system based on maximum entropy. In *Computational Linguistics and Intelligent Text Processing*, pages 143–153. Springer.

Yassine Benajiba, Mona Diab, Paolo Rosso, et al. 2008. Arabic named entity recognition: An svm-based approach. In *Proceedings of 2008 Arab International Conference on Information Technology (ACIT)*, pages 16–18.

Mona Diab. 2009. Second generation amira tools for arabic processing: Fast and robust tokenization, pos tagging, and base phrase chunking. In *2nd International Conference on Arabic Language Resources and Tools*.

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

Ralph Grishman and Beth Sundheim. 1996. Message understanding conference-6: A brief history. In *COLING*, volume 96, pages 466–471.

Nizar Habash and Fatiha Sadat. 2006. Arabic preprocessing schemes for statistical machine translation. In *NAACL*. ACL.

Shereen Khoja and Roger Garside. 1999. Stemming arabic text. *Lancaster, UK, Computing Department, Lancaster University*.

Wenhui Liao and Sriharsha Veeramachaneni. 2009. A simple semi-supervised algorithm for named entity recognition. In *Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing*, pages 58–65. Association for Computational Linguistics.

Xiaohua Liu, Shaodian Zhang, Furu Wei, and Ming Zhou. 2011. Recognizing named entities in tweets. In *ACL*, pages 359–367.

Slim Mesfar. 2007. Named entity recognition for arabic using syntactic grammars. In *Natural Language Processing and Information Systems*, pages 305–316. Springer.

Behrang Mohit, Nathan Schneider, Rishav Bhowmick, Kemal Oflazer, and Noah A Smith. 2012. 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*, pages 162–173. Association for Computational Linguistics.

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

David Nadeau, Peter Turney, and Stan Matwin. 2006. Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. *Advances in Artificial Intelligence*.

Mai Oudah and Khaled F Shaalan. 2012. A pipeline arabic named entity recognition using a hybrid approach. In *COLING*, pages 2159–2176.

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

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

Khaled Shaalan and Hafsa Raza. 2007. Person name entity recognition for arabic. In *Proceedings of the 2007 Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources*, pages 17–24. Association for Computational Linguistics.

Khaled Shaalan and Hafsa Raza. 2009. Nera: Named entity recognition for arabic. *Journal of the American Society for Information Science and Technology*, 60(8):1652–1663.

Peter D Turney, Patrick Pantel, et al. 2010. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37(1):141–188.