ANEAR: Automatic Named Entity Aliasing Resolution

Ayah Zirikly and Mona Diab

Department of Computer Science
The George Washington University
Washington DC, USA
{ayaz,mtdiab}@gwu.edu

Abstract. Identifying the different aliases used by or for an entity is emerging as a significant problem in reliable Information Extraction systems, especially with the proliferation of social media and their ever growing impact on different aspects of modern life such as politics, finance, security, etc. In this paper, we address the novel problem of Named Entity Aliasing Resolution (NEAR). We attempt to solve the NEAR problem in a language-independent setting by extracting the different aliases and variants of person named entities. We generate feature vectors for the named entities by building co-occurrence models that use different weighting schemes. The aliasing resolution process applies unsupervised machine learning techniques over the vector space models in order to produce groups of entities along with their aliases. We test our approach on two languages: Arabic and English. We study the impact of varying the level of morphological preprocessing of the words, as well as the part of speech tags surrounding the person named entities, and the named entities' distribution in the data set. We create novel evaluation data sets for both languages. NEAR yields better overall performance in Arabic than in English for comparable amounts of data, effectively using the POS tag information to improve performance. Our approach achieves an $F_{\beta=1}$ score of 67.85% and 70.03% for raw English and Arabic data sets, respectively.

1 Introduction

Named Entity Aliasing Resolution is the process where the different instances (aliases and variants) of an entity are detected and recognized as being referents to the same person within large collections of data. An example of this problem is shown in Figure 1 where each cluster contains several aliases for the same person (e.g. *Yasser Arafat, Abou Ammar*). The variation in name aliases can manifest as a difference in spelling (e.g. *Qaddafi, Gazzafi, Qadafi, Qazzafy*), difference in the name mention such as *Mohamed Hosni Mubarak*, vs. *Hosni Mubarak*, or by using a completely different alias such as *Abou Mazen* as an alternate for *Mahmoud Abbas*. Restricting this problem to aliases of famous people leads to a relatively easier resolution process since the aliases are typically publicly known. However, with the proliferation of web based data and social media, we note the pervasive use of aliases by ordinary people. Nowadays, the use of aliases and fake names is increasingly spreading among larger groups of people and becoming more popular due to political (terrorism, revolutions), criminal and privacy reasons. Hence, the ability to recognize and identify the different aliases of an

E. Métais et al. (Eds.): NLDB 2013, LNCS 7934, pp. 213–224, 2013.

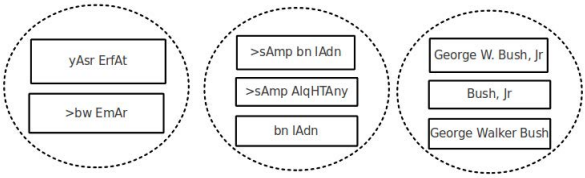

Fig. 1. Personal Named Entities and examples of possible aliases

entity improves the quality of information extracted (higher recall) by helping the entity linking and tracking, leading to better overall information extraction performance.

The NEAR task is relatively close to the Entity Mention Detection (EMD) task.[1,2] However they differ in several aspects. In NEAR there is no processing of pronominal mentions by definition. Moreover, the NEAR task, as defined for this paper, specifically focuses on detecting aliases for person named entities (PNE) and does not handle other NE types such as Organizations and Locations addressed in the EMD task. We should highlight, however, that there is nothing inherent in the NEAR task that bars it from processing other types of NEs. To date, most work in relating PNEs in documents relies on external resources, such as Wikipedia to provide links between aliases and PNE, thus confining the aliasing resolution task to famous people. In this paper, we build a system, Automatic NEAR (ANEAR), that is domain and language independent and does not rely on external knowledge resources. We use unsupervised clustering methods to identify and link the different candidate variants of an entity. We experiment with two languages, Arabic and English, independently. We empirically examine the impact of morphological processing on the feature space. We also investigate the usage of part of speech tag information in our models. Finally, we attempt to measure the effect of various value content modeling approaches on the system such as TF-IDF and co-occurrence frequency. ANEAR's best performance is $F_{\beta=1}$ score is 70.03% on Arabic compared to an $F_{\beta=1}$ score of 67.85% on the English data.

2 Automatic Name Entity Aliasing Resolution (ANEAR) Approach

The underlying assumption for ANEAR is that a person, regardless of his/her number of aliases, can be represented with a finite number of features that identifies him/her. These features encapsulate his/her interests, behaviors, writing style, background, spatial and temporal activities, etc.

The ANEAR system takes as input the unstructured text and generates a feature vector for every PNE as recognized in our data by a Named Entity Recognition (NER) system, i.e. this feature space models the profile for each PNE. The collection of feature vectors produces a Name Features Relatedness (NFR) matrix representing the vector space

[1] http://www.itl.nist.gov/iad/mig//tests/ace/ace07/doc/ace07-evalplan.v1.3a.pdf

[2] http://projects.ldc.upenn.edu/ace/docs/English-Entities-Guidelines_v6.6.pdf

model. We populate the NFR matrix with different values based on variable weighting schemes that reflect the relatedness scores. Subsequently, we apply unsupervised clustering algorithms to extract and group the different aliases and variants of an entity in one cluster. We experiment with two languages English and Arabic and use parallel data of the same size in order to compare and contrast performance cross-linguistically.

2.1 Building the Name Features Relatedness(NFR) Matrix

The selection of the features in conjunction with the relatedness scoring scheme has a significant impact on the performance of the clustering algorithm. The structure of the matrix is as follows: the row entries of the matrix are the PNEs, the dimensions are either bag of words (BOW) features or classes derived from them such as POS tags, and the feature values are some form of the co-occurrence statistic between the PNE and the feature instance.

2.1.1 Feature Dimensions. Our basic feature set is a BOW feature. We experiment with several possible tokenization levels for the words in the data collection: (i) LEX Inflected forms known as lexemes e.g. *babies* is a lexeme and contractions such as *isn't* are spelled out as *is not*; (ii) LEM Citation forms known as lemmas[3], *babies* is the lexeme and it would be reduced to the lemma *baby*, likewise the lexeme *is* becomes the lemma *be*. It is worth noting that for Arabic, a characteristic of the writing system is that words are typically rendered without short vowels and other pronunciation markers known as diacritics. For our purposes the LEM for Arabic will be the fully diacritized lemma, and the Lexeme, LEX is not diacritized. In order to identify if diacritization helps our process on the lexeme and the lemma levels, we explore a third word form in Arabic which is the diacritized lexeme DLEX. An example of a diacritized lexeme in Arabic is the DLEM *xaAmiso*,[4] *fifth*, and its undiacritized form is *xAms*.

Creating the vector space model for English and Arabic varies due to the nature of the two languages. Arabic has a much more complex morphological structure than English. Hence, as expected the number of lexeme dimensions for Arabic far exceeds that for English. Moreover, the lexeme to lemma ratio in Arabic is much higher in Arabic compared to English. We note that our Arabic data collection has 71910 diacritized lexemes compared to 67125 undiacritized lexeme and 38537 diacritized lemmas corresponding to a 6.65% and 46.41% reduction in the feature space for LEX and LEM, respectively, compared to DLEX in Arabic. For English the number of lexemes is significantly smaller for the same data collection size, 41317 lexemes corresponding to 32890 lemmas, representing a relatively smaller reduction in the feature space, going from LEX to LEM, of 20.4%.

[3] It should be noted that lemmas are also lexemes however they are a specific inflectional form that are conventionally chosen as a citation form, for example a typical lemma for a noun is the inflected 3rd person masculine singular form of the noun.

[4] All the Arabic used in this paper uses the Buckwalter transliteration scheme as described in `http://www.qamus.com`

2.1.1.1 Extended Dimensions In order to reduce the sparseness of the NFR matrix and add a level of abstraction, we augment the features space with part of speech (POS) tag features. Algorithm 1 explains the mechanism of generating the congregated POS features.

Data: ANEAR window_size x, POS window_size y[5], input dataset
for *every PNE per* $\in text\_win$ **do**
  if *distance(token, per)*$\leq y$ **then**
    $tags = tags \bigcup POS\_tag(token)$
  end
end
for *every tag*$\in tags$ **do**
  increment the frequency$_{features_vector(per)}$(class representative of POS tag)
end

Algorithm 1. Generate POS features

2.1.1.2 Feature Values are assigned based on one of the following metrics:

1. Co-occurrence Frequency (COF): PNE-feature co-occurrence frequency within a predetermined context window size of a sentence, SENT where the feature and the PNE co-occur in the same sentence, or a document, DOC, where the feature and the PNE co-occur in the same document. This results in either COF-SENT or COF-DOC.
2. Term Frequency-Inverse Document Frequency (TF-IDF): TF-IDF is calculated over the entire document collection. We have two settings varying the document size parameter for TF-IDF: (i) TF-IDF-DOC is based on using the entire collection of documents, and (ii) TF-IDF-PNE is based on constraining the document collection to those documents that mention the PNE. Both TF-IDF-DOC and TF-IDF-NE use the same equations as defined in 2 and 1 for calculating the feature values, however the former uses the entire document collection to calculate the values for DOC in the equations, while the latter is constrained to the document collection that mentions the PNE of interest, i.e. the vector row entry PNE in the matrix. Intuitively, both metrics capture the relative importance of the feature with respect to the PNE in a given document collection.

$$idf(feature,^{6}DOC) = log\frac{|DOCs|}{DOC \in DOCs : feature \in DOC} \quad (1)$$

$$tf(feature, DOC) = \frac{feature\_count(feature, DOC)}{max\{feature\_count(feature, DOC) : feature \in DOC} \quad (2)$$

3. Relative Rank Order (RRO): In this metric for the feature values, we abstract away from the absolute magnitude of the COF values or the TF-IDF values and we

Table 1. Sample NFR matrix illustrating the Feature Value (FV) Metrics COF-DOC values and their corresponding RRO values for the the various PNEs across 6 Lemma feature dimensions

	FV Metric	**president**	**chief**	**kill**	**assassin**	**Saudi**	**negotiation**
George Bush	*COF-DOC*	10	20	25	15	8	0
	RRO	4	2	1	3	5	0
Abu Ammar	*COF-DOC*	25	12	0	12	0	20
	RRO	1	3	0	3	0	2
Mahmoud Abbas	*COF-DOC*	20	11	8	1	0	35
	RRO	2	3	4	5	0	1
Abou Mazen	*COF-DOC*	24	16	5	2	0	30
	RRO	2	3	4	5	0	1
Yasser Arafat	*COF-DOC*	16	9	4	9	2	25
	RRO	2	3	4	3	5	1
G. W. Bush	*COF-DOC*	7	18	22	12	9	1
	RRO	5	2	1	3	4	6

replace them with their relative vector rank order value. Table 1 illustrates an example of the mapping between the COF-DOC values and the corresponding RRO values.[7]

2.1.2 Clustering and Retrieving the Different Groups of PNEs. We apply unsupervised clustering using the cosine similarity function across the feature vectors in order to produce the multiple groups of entities along with their aliases, i.e. grouping PNEs. Our chosen clustering approach takes as input the NFR sparse matrix and applies the Repeated Bisection clustering method that locally and globally optimizes the clustering solution C which contains multiple groups of entities conjoined with their instances.

$$C = \left\{ c : \; c = \bigcup_{PNE\, e} alias_e \right\} \tag{3}$$

3 Evaluation

3.1 Data and Preprocessing Tools

All of our experiments use the GALE Phase (2) Release (1) parallel dataset for English & Arabic.[8] We preprocessed the Arabic and English datasets in order to produce the NER tags, lexemes, lemmas and the Arabic diacritized lemmas. For all the

[7] We experiment with assigning a rank order value of 0 to the features that have a COF/TFIDF value of 0 versus, giving it the lowest rank order value in a given vector. We note that assigning missing features a value of 0 yielded significantly better results over ranking the missing features as the lowest rank order in the vector due to two factors: assigning the 0 features the lowest rank renders the actual rank variable across different vectors introducing significant noise, i.e. similar missing features will have different rank order values across different PNE row entries. The effect is exacerbated given the significant sparseness in the matrix.

[8] LDC2007E103. `(http://www.ldc.upenn.edu)`.

English preprocessing we use the Stanford CoreNLP toolset [1], for Arabic we use AMIRA by [2] for lexeme, diacritized lemma and undiacritized lemma generation. We use NIDA-ANER, the Arabic Named Entity Recognition by [3] to produce PNE tagged data. Figure 2 depicts the ANEAR processing steps.

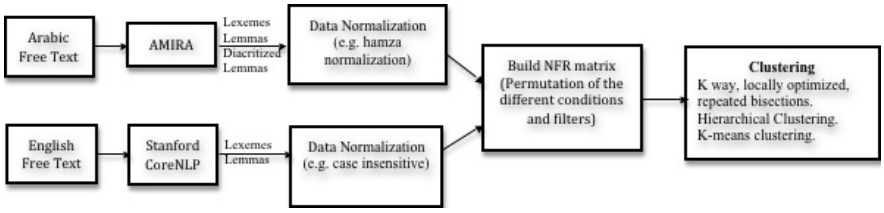

Fig. 2. ANEAR System Process

Due to the lack of annotated evaluation data for the aliasing resolution problem in Arabic and the limited evaluation data in English, we create our own English and Arabic evaluation data from the GALE dataset. Building the gold file comprises the following steps: a) Extract and list all the PNEs in the GALE dataset; b) In order to avoid singleton cases we set a unigram frequency threshold of ≥ 100 for each of the PNEs in order to be added to any of our clusters. This process yields an A list; c) Then we extract the transliterations of the PNEs based on string edit distance similarity measures for A; d) We then manually identify the aliases of the PNE in A in the dataset. The resulting gold standard file yields 26 PNE clusters in each language along with their respective aliases. The total number of PNEs in the Arabic set is 116 corresponding to 26 PNE clusters, and the total number of PNEs in English is 105 corresponding to 26 PNE clusters.

For automatic clustering, we use the CLUTO software package,[9] which employs multiple classes of k-way clustering algorithms that clusters low and high dimensional datasets with various similarity functions. CLUTO shows a robust clustering performance that outperforms many clustering algorithms such as K-means. We use the Repeated Bisection algorithm with default parameter settings. This clustering algorithm is a hard clustering algorithm. For clustering performance comparative reasons, we also use Matlab[10] implementations of the K-means and Hierarchical clustering algorithms.

3.2 Experimental Conditions

For each language, we have combinations of the following considerations. For the feature dimensions: (i) word tokenization level: Lexemes (LEX) vs. lemmas (LEM) vs. diacritized lexemes (DLEX) (the latter is only for Arabic). For the feature values, we have the following conditions: (i) simple co-occurrence frequency: COF-SENT and COF-DOC; (ii) TF-IDF-DOC and TF-IDF-NE; (iii) Rank Order with four settings:

[9] `http://glaros.dtc.umn.edu/gkhome/views/CLUTO`

[10] MATLAB and Statistics Toolbox Release 2009, The MathWorks, Inc., Natick, Massachusetts, United States.

RRO-COF-SENT, RRO-COF-DOC, and RRO-TFIDF-DOC, RRO-TFIDF-NE. We also have two feature sets: default bag of words, BOW, and BOW augmented with POS tag features, BOW+POS. Hence for English, this yields 2 word tokenization levels LEX/LEM * 8 feature value settings COF-SENT/COF-DOC/TF-IDF-DOC/TF-IDF-NE/RRO-COF-SENT/RRO-COF-DOC/RRO-TFIDF-DOC/RRO-TFIDF-NE amounting to 16 experimental conditions for each of the two feature settings BOW and BOW+POS, respectively. For Arabic, we have the following experimental conditions: 3 word tokenization levels LEX/LEM/DLEX *8 feature value settings COF-SENT/COF-DOC/TF-IDF-DOC/TF-IDF-NE/RRO-COF-SENT/RRO-COF-DOC/RRO-TF-IDF-DOC/RRO-TF-IDF-NE amounting to 24 experimental conditions for each of the two feature settings BOW and BOW+POS, respectively. Finally, we include the results of a naive baseline where the names are randomly assigned to one of 26 possible clusters, similar to our formulation of the problem, a PNE can only be assigned to one cluster (hard clustering).

3.3 Results

In Table 2, all the ANEAR conditions outperform the random baseline by a significant margin. ANEAR best results for English are obtained in the LEM_COF-DOC experimental setting achieving an $F_{\beta=1}$ score of 67.85% using the augmented POS features, and the best results for Arabic are achieved in the condition LEM_TF-IDF-DOC in the BOW+POS condition achieving an $F_{\beta=1}$=70.03%, with a narrow second condition LEX_TF-IDF-DOC with a score of $F_{\beta=1}$=69.58%.

In general with the BOW setting, the TF-IDF conditions outperform the comparative COF conditions. For example, in the English results, we note that LEX_TF-IDF-DOC|NE both outperform LEX_COF-SENT|DOC conditions (60.63% and 53.57% vs. 49.66% and 41.56%, respectively). Moreover, in the BOW setting, using RRO adversely impacts performance in both languages.

For both languages, The COF-DOC conditions outperform the COF-SENT conditions across the board. Also the TF-IDF-DOC conditions outperform the TF-IDF-NE conditions in the BOW setting, suggesting that narrowing the document collection extent is adverse to system performance.

For English, LEM conditions outperform LEX conditions except in the TF-IDF-DOC condition. However in the latter condition the difference between LEM and LEX conditions is relatively small (1%). In Arabic, the results are more consistent with LEM outperforming both LEX and DLEX in all the conditions, in the BOW setting.

Adding POS tag features has an overall positive impact on performance in English. In Arabic the story is quite different. The COF-SENT conditions in Arabic yield the worst results. But adding POS tag information to the other models seems to significantly improve performance.

For the Arabic experiments, under the BOW setting, the best F-score of 68.99% is obtained from the diacritized dataset (LEM) with TF-IDF-DOC. Using DOC provides better performance compared to SENT. Similarly to English results, adding POS tags to the feature space improves performance in both the LEX and LEM conditions, but not in the DLEX condition. This may be attributed to level of detail present in the DLEX forms combined with the detailed POS tag used. The best performing condition

Table 2. ANEAR $F_{\beta=1}$scores performance for both English and Arabic datasets under the different experimental conditions and feature settings, BOW and BOW+POS

Condition	English		Arabic	
	BOW	**BOW+POS**	**BOW**	**BOW+POS**
Random Baseline	31.96	31.96	31.16	31.16
LEX_COF-SENT	41.56	44.19	56.52	42.4
LEX_RRO-COF-SENT	39.46	45.18	54.43	43.25
LEM_COF-SENT	43.05	39.84	60.17	39.36
LEM_RRO-COF-SENT	43.99	46.22	53.14	42.93
DLEX_COF-SENT	-	-	60.29	42.9
DLEX_RRO-COF-SENT	-	-	52.66	39.44
LEX_COF-DOC	49.66	64.25	59.15	62.75
LEX_RRO-COF-DOC	47.88	65.01	57.33	60.33
LEM_COF-DOC	51.91	**67.85**	64.42	62.75
LEM_RRO-COF-DOC	48.17	66.52	56.77	60.87
DLEX_COF-DOC	-	-	65.83	63.16
DLEX_RRO-COF-DOC	-	-	56.94	63.28
LEX_TF-IDF-NE	53.67	65.12	60.66	64.25
LEX_RRO-TF-IDF-NE	46.55	63.82	53.51	65.64
LEM_TF-IDF-NE	57.41	64.36	67.3	65.83
LEM_RRO-TF-IDF-NE	47.09	63.82	49.93	60.87
DLEX_TF-IDF-NE	-	-	66.63	65.83
DLEX_RRO-TF-IDF-NE	-	-	49.45	60.87
LEX_TF-IDF-DOC	60.63	64.47	65.88	69.58
LEX_RRO-TF-IDF-DOC	36.05	62.62	40.26	63.12
LEM_TF-IDF-DOC	59.65	62.67	65.12	**70.03**
LEM_RRO-TF-IDF-DOC	40.52	62.74	40.6	62.08
DLEX_TF-IDF-DOC	-	-	68.99	66.76
DLEX_RRO-TF-IDF-DOC	-	-	41.19	62.08

yields an f-score of 70.03% in the LEM, TF-IDF-DOC setting. This is a significant improvement over the same condition setting without POS tag features which yielded an f-score of 65.12% only. It is worth noting that the POS tag set in Arabic is quite rich almost fully specifying the morphology of the word encoding significant semantic attributes unlike the English tag set that is purely syntactic. The emphasis on semantic features seems to be further corroborated by the noticeable improvement using LEM compared DLEX and LEX, leading to a more dense representation. Moreover more evidence comes from the fact that DLEX outperforms LEX in all the DOC conditions.

4 Discussion

4.1 Balancing the Data

We are cognizant of the unbalanced distribution of the aliases in the dataset within one cluster which highly affects the clustering performance. Hence, in addition to testing on

Data: The free text, Gold clusters
Result: A new redistribution of gold PNEs in the input text
for *every occurrence of a PNE* ***name*** *that is in the gold clusters* **do**
  cluster_id = get cluster ID of the input name
  with the use of uniformly distributed random number generator, retrieve randomly a member $new\_alias : new\_alias \in cluster_{cluster\_id}$
  replace $name$ with new_alias
end

Algorithm 2. Balancing and Resampling the dataset

the original dataset, we generate another balanced version that has a more normalized distribution based on the following approach:

When we balance the evaluation data, we observe an overall significant increase in absolute performance where the best condition LEM_COF-SENT yields an F-score of 96.05% for English compared to the best condition in Arabic of LEM_TFIDF-NE yielding an F-score of 96.45%.

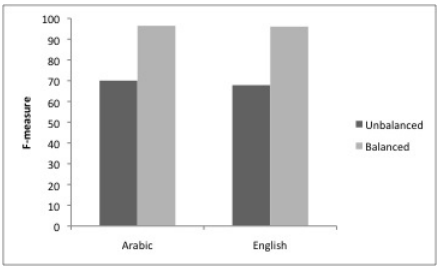

Fig. 3. ANEAR performance comparison between balanced and unbalanced Arabic and English datasets

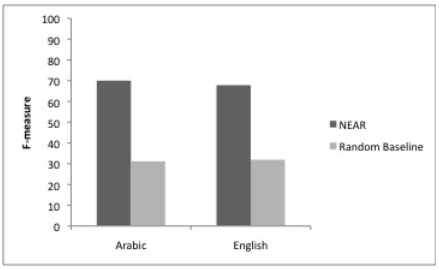

Fig. 4. Comparison between ANEAR and random baseline performance

Arabic shows more robust results and seems less affected (f-score = 70.03%) when compared to English (f-score = 67.85%). The more balanced distribution scheme adds a significant performance improvement ($\approx +25\%$) as shown in Figure 3. Based on the results, we generally notice that diacritized lexemes produce better performance, despite the higher feature dimensionality that yields a more sparse data set, yet decreasing the ambiguity results is a gain. Figure 3 contrasts ANEAR performance against a random baseline system with a gain of $\approx +39\%$ in Arabic and $\approx +30\%$ in English.

4.2 Alternate Clustering Algorithms

Additionally, we carry out a comparison assessment evaluation for our system against different clustering algorithms, namely, K-Means and Hierarchical clustering. Both K-Means and CLUTO Repeated Bisection require the number of clusters as an input parameter, and they yield their best performance under the same conditions. Whereas

Hierarchical clustering, though it does not require specifying the number of clusters as an input parameter, the number of clusters is automatically induced, it yields much poorer F-score results.

K-Means achieves the best performance under the condition DLEX_TF-IDF-NE (in Arabic) with an $F_{\beta=1}$score of 36.49%. On the other hand, Hierarchical clustering shows its best performance under the condition: LEX_COF-DOC with an $F_{\beta=1}$score of 21.38%. Figure5 shows a comparison among the different clustering algorithms when tested on balanced and unbalanced dataset.

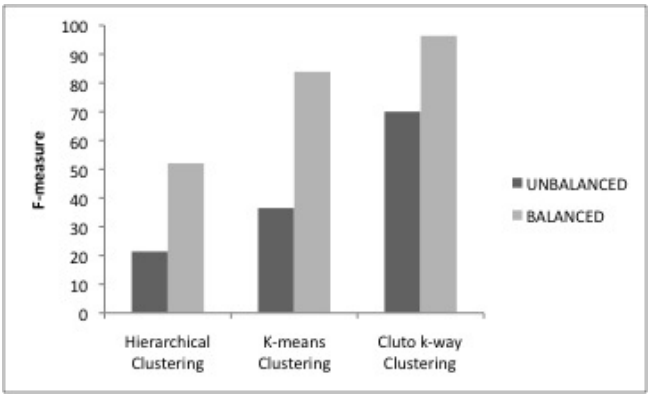

Fig. 5. Comparison among Hierarchical, K-means and CLUTO Repeated Bisection K-way Clustering when tested on the Arabic balanced and unbalanced datasets

5 Related Work

To date, most of the work related to the aliasing resolution problem has been mainly performed in the area of Named Entity Disambiguation, where two entities share the same name. Moreover, the NED task has typically focused on English since there are no annotated data sets for other languages. Our work employs unsupervised techniques to induce the PNE groups of name aliases while most work that we are aware of to date, uses predefined lists of PNEs and their corresponding aliases and used for training in a supervised manner. [4] proposed a framework for alias detection for a given entity using a logistic regression classifier that relies on a number of features such as co-occurrence relevance. Similarly, [5] presented a more complicated system that also relies on an input list of names and their aliases. They first retrieve a list of candidate aliases for a given entity using lexical patterns that introduce aliases, then they rank the set of retrieved aliases based on different factors: a) Lexical pattern frequency, b) Co-occurrence in anchor texts using different metrics such as TF-IDF and cosine similarity functions, and, c) Page counts of name-alias co-occurrence. [6,7] and [8] proposed a knowledge-based method that captures and leverages the structural semantic knowledge in multiple knowledge sources (such as Wikipedia and WordNet) in order to improve the disambiguation performance. Other disambiguation methods utilize ranked similarity measurements among entity-based summaries. [9,10]. [11] have used unsupervised

clustering algorithms on a rich feature space that is extracted from biographical facts. In PNE identification, [12] proposes a lexical pattern-based approach to extract a large set of candidate aliases from a web search engine. Then, a myriad of ranking scores (lexical pattern frequency, word co-occurrences and page counts on the web) are integrated into a single ranking function and fed into a support vector machines (SVM) to identify and predict aliases for a particular PNE.

Other contributions involved handling structured datasets such as Link Data Sets. [13] presented a hybrid probabilistic orthographic-semantic supervised learning model to recognize aliases.

Entity linking tackles a similar problem to NEAR where a name mention is mapped to an entry in a Knowledge Base (KB). Entity Linking relies heavily on Wikipedia pages to populate the KB and generates a dictionary that is used in name-variant mappings as illustrated in [14]. They integrate a number of features in order to choose the best mapping. These features include the surface forms, semantic links which assumes the availability of structured data and weighted bag of words features that are extracted from the Wikipedia documents. All of the above features assume that the entities to be resolved with their aliases are celebrities where Wikipedia reference them and their aliases.

Our approach provides a broader range of alias identification, since it does not rely on any lexical or string similarity properties. In addition, the identification process is executed offline with no dependence on external resources.

6 Conclusion

In this paper, we present a statistical, domain-independent aliasing resolution system, ANEAR. In building our system and exploring the search space, we experiment with different feature types and values and we measure their impact within two different languages Arabic and English. We note that employing semantically and syntactically oriented features helps performance. Also our results suggest that balancing the data set, namely the alias distribution, plays a role in improving performance. Our system is the first for ANEAR in Arabic. Our work results in annotated data sets for both Arabic and English.

Our best results on unbalanced Arabic and English datasets are $F_{\beta=1} = 70.03\%$ and $F_{\beta=1} = 67.85$, respectively.

Acknowledgments. This work is supported by the Defense Advanced Research Projects Agency (DARPA) BOLT program.

References

1. Finkel, J.R., Grenager, T., Manning, C.: Incorporating non-local information into information extraction systems by gibbs sampling. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL 2005, pp. 363–370. Association for Computational Linguistics, Stroudsburg (2005)

2. Diab, M.: Second generation tools (amira 2.0): Fast and robust tokenization, pos tagging, and base phrase chunking. In: Choukri, K., Maegaard, B., eds.: Proceedings of the Second International Conference on Arabic Language Resources and Tools. The MEDAR Consortium, Cairo (2009)
3. Benajiba, Y., Diab, M.T., Rosso, P.: Arabic named entity recognition: A feature-driven study. IEEE Transactions on Audio, Speech & Language Processing 17(5), 926–934 (2009)
4. Jiang, L., Wang, J., Luo, P., An, N., Wang, M.: Towards alias detection without string similarity: an active learning based approach. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2012, pp. 1155–1156. ACM, New York (2012)
5. Bollegala, D., Matsuo, Y., Ishizuka, M.: Automatic discovery of personal name aliases from the web. IEEE Trans. on Knowl. and Data Eng. 23(6), 831–844 (2011)
6. Han, X., Zhao, J.: Structural semantic relatedness: A knowledge-based method to named entity disambiguation. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 50–59. Association for Computational Linguistics, Uppsala (2010)
7. Cucerzan, S.: Large-scale named entity disambiguation based on wikipedia data. In: Proceedings of EMNLP-CoNLL, vol. 2007, pp. 708–716 (2007)
8. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using wikipedia-based explicit semantic analysis. In: IJCAI 2007: Proceedings of the 20th International Joint Conference on Artifical Intelligence, pp. 1606–1611. Morgan Kaufmann Publishers Inc., San Francisco (2007)
9. Bagga, A., Baldwin, B.: Entity-based cross-document coreferencing using the vector space model. In: COLING-ACL, pp. 79–85 (1998)
10. Bagga, A., Biermann, A.W.: A methodology for cross-document coreference. In: Proceedings of the Fifth Joint Conference on Information Sciences (JCIS 2000), pp. 207–210 (2000)
11. Mann, G.S., Yarowsky, D.: Unsupervised personal name disambiguation. In: Daelemans, W., Osborne, M. (eds.) Proceedings of CoNLL-2003, pp. 33–40. Edmonton, Canada (2003)
12. Bollegala, D., Matsuo, Y., Ishizuka, M.: Automatic discovery of personal name aliases from the web. IEEE Trans. Knowl. Data Eng. 23(6), 831–844 (2011)
13. Hsiung, P., Moore, A., Neil, D., Schneider, J.: Alias detection in link data sets. Master's thesis, Technical Report CMU-RI-TR-04-22 (March 2004)
14. Charton, E., Gagnon, M.: A disambiguation resource extracted from wikipedia for semantic annotation. In: LREC, pp. 3665–3671 (2012)
15. Chen, Y., Martin, J.: Towards robust unsupervised personal name disambiguation. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pp. 190–198. Association for Computational Linguistics, Prague (2007)
16. Sutton, C., Mccallum, A.: Introduction to Conditional Random Fields for Relational Learning. MIT Press (2006)