Entity Network Extraction Based on Association Finding and Relation Extraction

Ridho Reinanda[1,2], Marta Utama[1], Fridus Steijlen[2], and Maarten de Rijke[1]

[1] ISLA, University of Amsterdam, The Netherlands
{r.reinanda,derijke}@uva.nl, marta.utama@gmail.com
[2] Royal Netherlands Institute of Southeast Asian and Caribbean Studies
steijlen@kitlv.nl

Abstract. One of the core aims of semantic search is to directly present users with information instead of lists of documents. Various entity-oriented tasks have been or are being considered, including entity search and related entity finding. In the context of digital libraries for computational humanities, we consider another task, network extraction: given an input entity and a document collection, extract related entities from the collection and present them as a network. We develop a combined approach for entity network extraction that consists of a co-occurrence-based approach to association finding and a machine learning-based approach to relation extraction. We evaluate our approach by comparing the results on a ground truth obtained using a pooling method.

1 Introduction

Today's increasing digitization and curation of humanities content in digital libraries gives rise to a new and interesting set of opportunities. In computational humanities, researchers are particularly interested in applying computational methods and algorithms to gain insight from this kind of data [16]. One interesting and urgent problem is extracting and analyzing networks of entities from unstructured, possibly noisy text such as (archival) newspaper articles. Recognizing such entities (person, organization, or location) and discovering how they are connected to each other benefits computational humanities researchers asking questions about network and entities, for example in understanding the network of an elite politician and its dynamics [9].

We view entity network extraction task as a form of semantic search. Our working hypothesis is that having entities and related entities presented in the form of a network is more useful than returning a large list of documents and forcing users to go through each and every one of them to manually identify the connections. For our purposes a network is a graph with a main entity together with a set of related entities as nodes, with edges connecting these nodes. A connection between two nodes denotes that there is a relationship between these two entities according to evidence found in the text. In our computational humanities application scenario, our users use a manually constructed English corpus of newspaper articles about Indonesia collected over a 10 year period. This amounts to 140,263 articles, mostly consisting of politics and economy articles. Fig. 1 shows (part of) an entity network automatically extracted from the corpus. The query entity is "BJ Habibie," a former president of Indonesia. Because the query entity

T. Aalberg et al. (Eds.): TPDL 2013, LNCS 8092, pp. 156–167, 2013.

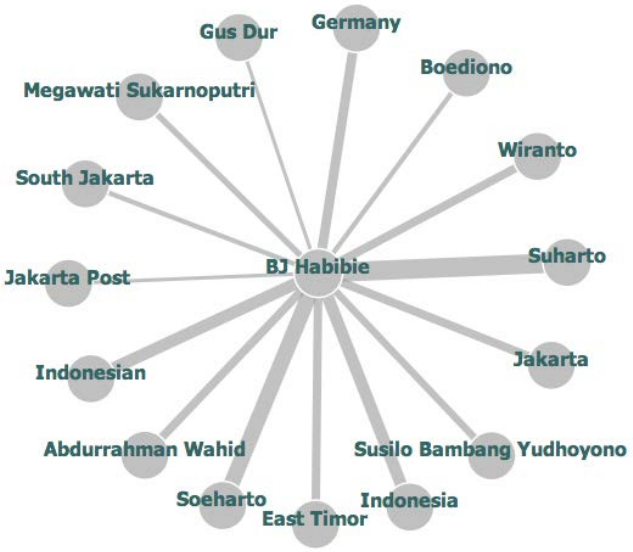

Fig. 1. A sample network retrieved in response to "BJ Habibie" as query entity. The thickness of the links depicts the association strength as represented in the document collection.

is a popular person, he is related to many other entities in the text. We rank the entity relations based on a scoring method, and build the network from top ranked entities only. We use an English-language corpus with Indonesian politics as the primary topic, but our approach also works on other languages with minor changes in the pipeline (utilizing respective linguistic tools). Our approach does not rely on domain-specific pattern extraction, so it is adaptable to other topics or domains as well.

The closest benchmarking task to our proposed task is the related entity finding (REF) task that was considered at TREC 2009, 2010 and 2011 [2]. Related entity finding works as follows: given a source entity, a target page, a narration of the relation of interest, one has to give a ranked list of entities and their home page that engage in this relation with the source. Our task is different from the REF task in the sense that we only have the names of the entities; no sample homepage, and no narration. Furthermore, we are not interested in a single specific relation, but in all possible relationships.

In this paper we address the task of extracting an entity network from text in two ways: (1) by discovering associations between entities through statistical or information-theoretic measures, and (2) by performing relation extraction and building a network using the relationships discovered. We contrast these two approaches and also consider a combination of the two types of approach based on pairwise learning-to-rank [13].

2 Related Work

Entity Network Extraction as Semantic Search. Previous research has dealt with extracting various kinds of network from document collections. Referral Web [14] takes a person name as input and finds people related to this person on the Web by using an

external search engine. Referral Web uses the number of pages where two person names co-occur to measure the degree to which they are related.

Merhav et al. [17] perform extraction of relational networks of entities from blog posts. This is done by first creating entity pairs, clustering those entity pairs, and later labeling these clusters with the nature of the relationship. Elson et al. [7] extract social networks from literary fiction. The networks are derived from dialogue interactions, thus the method depends on the ability to determine whether two characters are in a conversation. Their approach involves name chunking, quoted speech attribution, and conversation detection. Tang et al. [21] extract social networks of academic researchers. After entities are identified and disambiguated, they provide a shortest-path search mechanism that links the researchers and their publications as a network.

Association Measures. Association measures can be used to describe the relationship between two words or concepts. There are various ways to measures associations or relatedness. We distinguish between the following types: frequency-based, distance-based, distributional similarity/feature-based, and knowledge-based measures.

Frequency-based measures rely on the frequency of word co-occurrences and the (unigram) frequency of each word. These include measures that are derived from probability theory or information theory, for example Chi-Square, Pointwise Mutual Information, and Log Likelihood Ratio [6]. Distance-based measures rely on the distance between words in the text. Co-dispersion, introduced in [22], is one such measure.

Feature-based or distributional similarity measures describe the relatedness between two words or concepts based on the distribution of words around them. These are measures based on extracting a number of features for each entity, and then comparing the feature vectors for different entities. One example is by using cosine similarity to determine the relatedness of two entities based on linguistic features, such as neighboring words, part-of-speech tag, etc. [6]. Knowledge-based measures are measures that use on ontology, thesaurus, or semantic network to determine the relatedness between words or concepts [18].

Relation Extraction. In relation extraction, we want to extract relations between entities such as persons, organizations, and locations. *Supervised* methods view the relation extraction task as a classification task. Features are extracted from entity pairs and a classifier is trained to determine whether a pair of entities is related. There are various groups of methods: feature based methods, in which syntactic and semantic features are extracted from the text, and string kernel methods, where the whole string is passed as a feature and string kernel functions are used to recognize the richer representations of the structure within the strings to determine whether two entities are in a relation.

Semi-supervised methods are often based on pattern-based extraction algorithms. The core idea is bootstrapping, in which one tries to extract patterns iteratively, using newly found patterns to fuel later extraction steps. DIPRE [4] starts with a small set of entity pairs; the system then tries to find instances of those seeds. With newly found instances, the relation is generalized. Snowball [1] uses the same core idea. Snowball starts with a seed set of relations and attaches confidence scores to them; it uses inexact matching to cope with different surface structures. TextRunner [8] learns relations, classes, and entities from text in a self-supervised fashion. The system starts by

generating candidate relations from sentences, then uses constraints to label candidates as positive or negative examples to feed a binary classifier.

Since labeling and annotating a corpus to create relation examples is an expensive and time-intensive procedure, there is increasing attention for *unsupervised* or *weakly-supervised* approaches to relation extraction. With distant supervision [19], indirect examples in the form of relations from a knowledge base such as Freebase and DBPedia are used. From these relation tuples, instances of relations in the form of sentences in the corpus are searched. From these sentences, text features are extracted that are then used to train classifiers that can identify relations.

Our work differs from the related work described above in the following important ways. Firstly, in building the network, we also look at measures to determine the score of the related entities. Secondly, we experiment with alternative association measures, i.e., distance-based ones. Thirdly, while relation extraction methods usually train a specific classifier for each predefined relation type, we train a generic relation classifier on linguistic features. To the best of our knowledge, we are the first to consider combining association finding and relation extraction to extract an entity network from text.

3 Method

Task Description. The task of network extraction is as follows: given a corpus and an input entity as a query, we must return a list of related entities, along with scores that can be used to rank them. The scores can be used for visualization purposes, and can be interpreted as the strength of association between the entities, or number of pieces of evidence supporting an extracted connection.

Pipeline. In the preparation stage, we enrich each document with linguistic annotations. We perform the following types of linguistic processing: tokenization, part-of-speech tagging, sentence splitting, constituency parsing, and named entity recognition with the Stanford NLP tools [15]. We later construct an index out of these documents and their linguistic annotations.

Our main pipeline consists of the following steps: (1) query construction, (2) document selection, (3) entity extraction, (4) candidate scoring, and (5) candidate ranking.

(1) Query Construction. For each query entity e, we construct the query q, a phrase query that will be used in searching the index.

(2) Document Selection. For retrieval purposes in the search step, we use Lucene,[1] which combines a boolean model and vector space model. After obtaining the search results, we use all of the returned documents in the next step.

(3) Entity Extraction. For every document in the search result, we extract pairs of entities (x, y) that co-occur within the same sentence. We then filter these pairs of entities, to only consider pairs that contain the query entity e.

In filtering the pair of entities, we follow the rule-based inexact matching scheme used in expert finding [3], but we adapt the rules to suit our task:

[1] `http://lucene.apache.org`

- EXACT MATCH returns a match if x is mentioned exactly the same as query entity e.
- LAST NAME MATCH returns a match if x is the last name of the query entity e.
- FIRST NAME MATCH returns a match if x is the first name of the query entity e.

(4a) Candidate Scoring – Association Measure. A score is assigned for each entity pair based on association measures. We compute the association strength by several *frequency based measures*: pair frequency, pointwise mutual information (PMI), and Jaccard. In the following equations, $f(x, y)$ denotes the frequency of two entities appearing together in the same sentence, $f(x)$ is the unigram frequency of entity x within the set of selected documents, and $f(y)$ is the unigram frequency of entity y within the set. Pair frequency is computed as follows: $PF(x, y) = f(x, y)$. Pointwise mutual information is computed as follows: $PMI(x, y) = \log \frac{f(x,y)}{f(x)f(y)}$. The Jaccard measure is computed as follows: $Jaccard(x, y) = \frac{f(x,y)}{f(x)+f(y)-f(x,y)}$. Both document-level and sentence-level frequency are used as evidence in counting the frequency. With document-level frequency as evidence, $f(x, y)$ is basically the document frequency of entity pairs.

We also experiment with *distance-based measures*, first by simply using the average distance of two entities. Here distance means the number of tokens separating two entities. With M denoting mean, we define the *inverse mean distance* (IMD) as follows: $IMD(x, y) = \frac{1}{M(dist_{xy1},\ldots,dist_{xyn})}$, where $dist_i$ is the linear word distance at the pair occurrence i.

An alternative to linear word distance is dependency distance. To get a dependency distance, we first need to perform dependency parsing [10] on sentences containing the entity pair. The result of this parsing is a dependency tree. Entities are not stored in a single node in a parse tree, but broken down into component words. We define dependency distance as the number of edges between the head word of entity x to the head word of entity y. We find the shortest path between these two head word nodes, and use the number of edges as distance. We then simply subsitute dependency distance as $dist$ in the previous equation to compute the dependency-based IMD.

Based on the preliminary observation that simply using pair frequency performs quite well, we propose the following measure: $PF.IMD(x, y) = PF(x, y) \times IMD(x, y)$. This measure takes into account both frequency and average distance. The intuition behind this is that a good relation will spread across a lot of documents with small dependency distance.

(4b) Candidate Scoring – Relation Extraction. We use sentences containing the pairs of entities as text snippets. We extract the following features from each text snippet: named entity types, dependency distance, linear distance, typed dependencies (conjunction, noun modifier, or preposition), dependency trigram/bigram, and punctuation type between entities. Sentence level features are also extracted: number of tokens, the presence of quotes, and number of entities within the sentences. We avoid using lexical features in order to have a domain-independent, generic classifier.

We use a portion of our ground truth to train and tune a SVM classifier [20]. For every pair of entities that is extracted, we run the classifier to determine whether their snippets describe that the two entities are related. The snippets that are classified as

Table 1. Entity network extraction methods considered in the paper

Method	Description
pf-doc	Document-level pair frequency
pmi-doc	Document-level PMI
pf-sen	Sentence-level pair frequency
pmi-sen	Sentence-level PMI
jaccard-doc	Document-level Jaccard
jaccard-sen	Sentence-level Jaccard
imd-lin	Inverse mean distance, linear
imd-dep	Inverse mean distance, dependency
pf-doc.imd-dep	Document-level PF.IMD, dependency
pf-sen.imd-dep	Sentence-level PF.IMD, dependency
rel-conf	Relation confidence
rel-support	Relation support
rel-conf.rel-support	Relation confidence.support
ensemble-all	Ensemble of all methods
ensemble-freq	Ensemble of frequency methods
ensemble-dist	Ensemble of distance methods
ensemble-freq.dist	Ensemble of frequency and distance methods
ensemble-rel	Ensemble of relation extraction methods
ensemble-top-4	Ensemble of top 4 methods from feature selection
ensemble-top-6	Ensemble of top 6 methods from feature selection
ensemble-top-8	Ensemble of top 8 methods from feature selection
ensemble-top-10	Ensemble of top 10 methods from feature selection

correct relations will serve as $support$ instances to the relation. We score the entity pairs based on how many support instances remain after the classification. We also calculate the $confidence$ score of a pair, defined as the number of snippets detected as relations over all the snippets extracted containing the pair. We define another score as combination: $support.confidence$.

(5) Candidate Ranking. We simply rank entity pairs based on their score.

Combination Methods. As we will see below, the network extraction methods that we consider behave quite differently. Because of this, we also experiment with learning to rank for combining rankings produced by various methods. Specifically, we use RankSVM [13], a pairwise learning to rank algorithm. Scores from various network extraction methods are used to build an ensemble ranking model. We try different combinations of ensembles. First, training an ensemble using scores from all methods, and also ensembles built from each family of methods. We also experiment with ensembles based on automatic feature selection. We use a filtering approach, ranking features by importance, using randomized trees [11]. Randomized regression trees are built from subsamples of the training data. Feature importance is computed based on the number of times a feature is selected as decision node in the randomized trees [20]. We use the top 4, 6, 8, and 10 features from this feature selection step to build our ensembles.

Network Extraction Methods Compared. All in all, we consider the methods listed in Table 1 for extracting networks.

4 Experimental Setup

Research Questions. We aim to answer the following research questions. (RQ1) How do the methods based on association measures and relation extraction compare? (RQ2) Can we combine these various scoring methods in an ensemble to improve the performance? (RQ3) How does performance differ across different queries?

Dataset. We use a corpus manually constructed by social historians, from web articles during the period between 2000 and 2012.[2] The corpus contains 140,263 articles about Indonesia and South East Asia. These are mainly news articles from English language media based in Indonesia such as Jakarta Post and Jakarta Globe. Some articles from international media such as The Washington Post and The New York Times are also included. The articles are from diverse topics: politics, economy, cultural events, etc. Some of the named entities of the type organization and location appear in the their English version. An example of this case is "Badan Intelijen Negara" (BIN), which appears in the text both as "BIN" and "State Intelligence Agency."

Ground Truth. We prepare our ground truth by using a pooling strategy (similar to TREC [12]). We select 35 query entities that are known to occur in our corpus, run all entity network extraction methods listed in Table 1 and pool the top 10 related entities from each method. In the assessment step, pairs (query entity, related entity) are presented to three assessors (domain experts) along with supporting text snippets. The assessors' task is to decide whether the two entities are directly related based on the text snippets containing the pair. The assessors are not given a strict definition of a relation. In case of disagreement, the majority vote determines the final assessment. We reach 80 percent average pairwise agreement between the assessors, with a kappa value of 0.60.

Evaluation Metrics and Significance Testing. We use recall, precision and F-measure as a way to evaluate the performance of our entity network extraction methods. In this task, recall is the fraction of correct relations retrieved over all relations in our ground truth. Precision is the fraction of correct relations over the retrieved relations. We mainly look at the performance in the top ten and thirty entities returned. For significance testing, we use a paired t-test with $\alpha = 0.05$.

5 Results

We run our entity network extraction approach on the query entities with various scoring methods. Table 2 shows the results of extracting the top-10 and 30 related entities.

Methods Comparison. To answer RQ1, we look at the performance of the non ensemble methods. Overall, we can see that `pf-doc`, simply counting the number of documents in which the pair of entities co-occur, already provides a decent performance. Using the sentence count, `pf-sen`, further improves the performance. The `Jaccard` measures, both at the document and sentence count, perform slightly worse than `pf`. The `pmi-doc` and `pmi-sen` methods both perform significantly worse than the baseline.

[2] Access to the dataset and ground truth can be facilitated upon request.

Table 2. Results of the entity network extraction methods at top-10 and top-30 related entities. Significance is tested against the baseline with $\alpha = 0.05$.

Method	R@10	P@10	F@10
pf-doc (baseline)	0.506	0.544	0.478
pmi-doc	$0.365^{\blacktriangledown}$	$0.321^{\blacktriangledown}$	$0.295^{\blacktriangledown}$
pf-sen	0.519	0.558	0.491
pmi-sen	$0.328^{\blacktriangledown}$	$0.309^{\blacktriangledown}$	$0.281^{\blacktriangledown}$
jaccard-doc	0.520	0.529	0.468
jaccard-sen	0.483	0.529	0.460
imd-lin	0.434	$0.355^{\blacktriangledown}$	$0.350^{\blacktriangledown}$
imd-dep	0.425	$0.366^{\blacktriangledown}$	$0.347^{\blacktriangledown}$
pf-doc.imd-dep	0.516	0.515	0.461
pf-sen.imd-dep	0.519	0.524	0.465
rel-conf	$0.365^{\blacktriangledown}$	$0.326^{\blacktriangledown}$	$0.312^{\blacktriangledown}$
rel-support	0.489	0.501	0.452
rel-conf.rel-support	$0.443^{\blacktriangledown}$	$0.429^{\blacktriangledown}$	$0.398^{\blacktriangledown}$
ensemble-all	**0.569**	**0.564**	**0.507**
ensemble-freq	0.544	0.552	0.490
ensemble-dist	0.504	0.498	0.447
ensemble-rel	0.544	0.541	0.486
ensemble-freq.dist	0.470	$0.475^{\blacktriangledown}$	0.431
ensemble-top-4	0.409	$0.315^{\blacktriangledown}$	$0.321^{\blacktriangledown}$
ensemble-top-6	0.439	$0.349^{\blacktriangledown}$	$0.351^{\blacktriangledown}$
ensemble-top-8	0.548	0.535	0.484
ensemble-top-10	0.555	0.549	0.494

Method	R@30	P@30	F@30
pf-doc (baseline)	0.775	0.324	0.435
pmi-doc	$0.613^{\blacktriangledown}$	$0.245^{\blacktriangledown}$	$0.333^{\blacktriangledown}$
pf-sen	0.785	0.329	0.441
pmi-sen	$0.609^{\blacktriangledown}$	$0.241^{\blacktriangledown}$	$0.327^{\blacktriangledown}$
jaccard-doc	0.763	0.318	0.427
jaccard-sen	0.763	0.323	0.431
imd-lin	$0.670^{\blacktriangledown}$	$0.257^{\blacktriangledown}$	$0.354^{\blacktriangledown}$
imd-dep	$0.685^{\blacktriangledown}$	$0.268^{\blacktriangledown}$	$0.367^{\blacktriangledown}$
pf-doc.imd-dep	0.803	0.334	0.449
pf-sen.imd-dep	0.815	0.342	0.459
rel-conf	$0.712^{\blacktriangledown}$	$0.277^{\blacktriangledown}$	$0.381^{\blacktriangledown}$
rel-support	0.795	0.332	0.446
rel-conf.rel-support	0.777	0.321	0.433
ensemble-all	$0.822^{\blacktriangle}$	0.343	$0.461^{\blacktriangle}$
ensemble-freq	0.772	0.321	0.431
ensemble-dist	0.800	0.333	0.448
ensemble-rel	**0.825**$^{\blacktriangle}$	**0.346**$^{\blacktriangle}$	**0.465**$^{\blacktriangle}$
ensemble-freq.dist	0.788	0.328	0.442
ensemble-top-4	$0.685^{\blacktriangledown}$	$0.262^{\blacktriangledown}$	$0.362^{\blacktriangledown}$
ensemble-top-6	$0.703^{\blacktriangledown}$	$0.271^{\blacktriangledown}$	$0.374^{\blacktriangledown}$
ensemble-top-8	$0.818^{\blacktriangle}$	0.341	0.459
ensemble-top-10	$0.820^{\blacktriangle}$	0.342	$0.460^{\blacktriangle}$

PMI yields the worst performance compared to all other methods. When we look at the actual relations returned by `pmi-doc` and `pmi-sen`, we find that it is prone to extracting rare co-occurrences of entities. As a consequence, errors in the preprocessing stage (e.g., named entity recognition errors) sometimes appear in the results. Distance-based methods also perform worse than the baseline. Relying on distance alone, two entities that only appear once within close distance can easily be favored over the ones that appear more often.

We take a closer look by comparing the top-10 results of `pf-doc` and `imd-dep` on query entity "BJ Habibie." In Table 3 correctly related entities are shown in bold face.

On this particular query, `pf-doc` clearly outperforms `imd-dep`. Almost all of the non-related entities retrieved by `imd-dep` in the table appear with the query entity in the same sentence as *enumerations* (e.g., listings of people attending a particular event). In a dependency parse tree, this type of co-occurrence will appear with dependency distance of 1, with *conjunction* as the dependency type. It is interesting to note that by using average distance instead of frequency, we successfully retrieve relations that do not occur often in the text. The two relations: "IPTN" (company founded by BJ Habibie), and "Watik Pratiknya" (a friend of BJ Habibie) are the kind of relations that are less frequently present in our corpus, since news articles are more likely to describe event-based stories instead of giving description of one's family or friends.

Table 3. Comparing `pf-doc` and `imd-dep`

pf-doc	imd-dep
Suharto	Taufik Kiemas
Soeharto	Wahid
Indonesian	Megawati Soekarnoputri
Indonesia	**IPTN**
Germany	Emil Salim
Abdurrahman Wahid	**Watik Pratiknya**
Wiranto	Sudi Silalahi
East Timor	Soehardjo
Jakarta	Xanana Gusmao
Susilo Bambang Yudhoyono	Sarwono Kusumaatmadja

As we have seen, replacing frequency by distance has its own advantages and disadvantages. We proceed to look at the performance of our proposed method `pf.imd`, which combines frequency and distance. This combination yields some improvement over the baseline at top-30 results, but the improvement is not significant.

With `rel-conf`, the relations that are detected by the machine learning method, but only found in one sentence, can outweigh relations that appear in many sentences. This explains why `rel-support` has a better performance, even outperforming both `pf-doc` and `pf-sen` for the top-30 results. The method `rel-support`, which can be viewed as a filtered version of `pf`, classifies text snippets before counting the frequency. This provides a more reliable way of counting the pair frequency. However, when we see the per-query results, the classifier does not always work, leading to a lower average performance compared to `pf-doc` and `pf-sen` (for the top-10 results).

Next, we contrast the results of a relation extraction method, `rel-support` with `pf-doc`, again for the query "BJ Habibie." The relations are listed in Table 4. For this query, the filtering effect of the relation extraction classifier manages to improve the results. The resulting ranking introduces three new entities (all related) and pushes out one non-related entity.

Table 4. Comparing `rel-support` with `pf-doc`

pf-doc	rel-support
Suharto	**Suharto**
Soeharto	Abdurrahman Wahid
Indonesian	**Indonesian**
Indonesia	**Megawati Soekarnoputri**
Germany	**Soeharto**
Abdurrahman Wahid	**Germany**
Wiranto	**Susilo Bambang Yudhoyono**
East Timor	**Boediono**
Jakarta	**ICMI**
Susilo Bambang Yudhoyono	**Golkar**

Ensemble Methods. To answer RQ2, we contrast the results of our ensemble methods against the non-ensemble ones. Table 2 shows that most ensemble methods give improvements over the baseline. Indeed, the overall best performance is achieved using ensemble methods. The improvements are statistically significant at top 30 related entities (`ensemble-all` and `ensemble-rel`, and `ensemble-top-10`). Simply using all of the methods in one ensemble can give a good performance. Ensembles of methods within the same family do not perform as well as combining method from various families. An exception to this is `ensemble-rel`, which only combines relation extraction methods scores.

Interestingly, the tree-based feature selection returns the following as top-6 features: `imd-lin`, `jaccard-sen`, `relation-conf`, `jaccard-sen`, `pmi-sen`, and `imd-dep`. Using these top-4 and top-6 features in an ensemble results in poor performance. As we observed above, three of these scoring methods are among the worst performing methods, thus combining them without adding (many) other scoring functions reinforces the weaknesses.

Score Differences between Entities. To answer RQ3, we average the performance of all methods on each query. As shown in Fig. 2, the performance varies. Some entities appear frequently in the dataset, therefore having more possible candidates and more possible types of context and relations. However, there does not seem to be a direct correlation with entity network extraction performance.

What went wrong with the worst performing queries? The person in query-24, "J Kristiadi," is a political observer. Most sentences mentioning him in the text are statements containing his observation about other entities, while only two describing actual relations to his affiliations. On this extreme case, most methods fail. For query-26, most of the snippets consist of mentions of the query with other entities in the form of enumerations. The snippets of query-29 also contain speech statements about other entities, along with invalid snippets created due to sentence splitting errors.

As shown in Fig. 2, query-11 has the highest average performance. The person in query-11, "Edy Harjoko," is a military commander. Most snippets in the text mention his rank or role in the organization (i.e., "TNI Chief of General Affairs Edy Harjoko"). There is almost no direct/indirect speech found in the snippets of this query. The snippets of query-23 also consist of a lot "head of" and "founder of" mentions. The next best performing query contains a lot of snippets in the form of appositions (e.g., "who founded ..."). Overall, we can say that these queries have more reliable snippets.

Error Analysis. We further analyze the errors made by most methods. In particular, we look at the bottom-10 query entities for which the worst performance is observed. By inspecting the supporting text snippets, we discover several types of error, mostly caused by the type of sentence that are used to extract the co-occurrence.

One of the most common cases is sentences containing *indirect/direct speech*, in which one entity mentions other entities. The fact that one entity mentions another entity does not necessarily mean that they have a direct connection. The low performing queries tend to have more of this type of sentence than other queries, as we have shown with query-24.

Another common case of errors are *enumerations*. As we have described above, enumerations of entities do not necessarily mean that the entities enumerated are related.

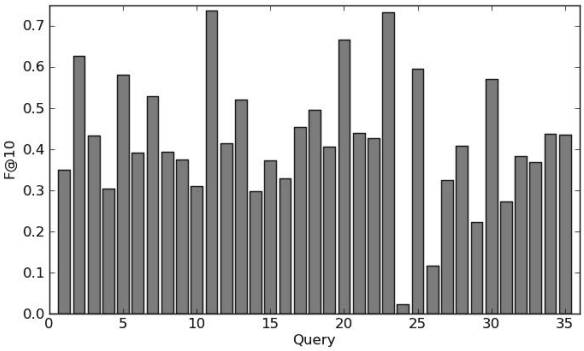

Fig. 2. Extraction performance per query (in F@10)

We observe that in our document collection most enumerations are ad-hoc, i.e., listing a number of entities that attend a certain event. When the text snippets returned for a query entity contain many enumerations, we tend to get a lower performance.

6 Conclusions

Today, more humanities content are archived and made available in digital libraries. We have presented the task of entity network extraction from text that can be applied to these types of contents. The task is studied in the context of a computational humanities application scenario. Our approach introduces an information retrieval pipeline that involves document search, entity extraction, and entity pairs scoring based on multiple scoring functions. We explore various methods for scoring extracted entity pairs, based on co-occurrences or relation extraction. In our experiments, we find that these methods display different behaviors. Combining them in a learning to rank ensemble successfully improves the performance.

As to future work, upon analyzing the results, we have discovered common errors related to certain sentence types that affect most methods' performance. Detecting indirect/direct speech as well as enumerations, and automatically filtering them out, is an interesting next step to improve the effectiveness of our approaches.

Additionally, to help users of the extracted networks interpret and contextualize the results, we aim to explore the usefulness of automatically linking the newspaper archive from which the networks have been extracted to other archives, similar to [5].

Acknowledgments. This research was supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreements nr 258191 (PROMISE Network of Excellence) and 288024 (LiMoSINe project), the Netherlands Organisation for Scientific Research (NWO) under project nrs 640.004.802, 727.011.-005, 612.001.116, HOR-11-10, the Center for Creation, Content and Technology (CCCT), the BILAND project funded by the CLARIN-nl program, the Dutch national program COMMIT, the ESF Research Network Program ELIAS, the Elite Network Shifts project funded by the Royal Dutch Academy of Sciences (KNAW), as well as the Netherlands eScience Center under project number 027.012.105.

References

[1] Agichtein, E., Gravano, L.: Snowball: extracting relations from large plain-text collections. In: DL 2000, pp. 85–94. ACM, New York (2000)

[2] Balog, K., Serdyukov, P., de Vries, A.P.: Overview of the TREC 2011 entity track. In: TREC 2011 Working Notes. NIST (2011)

[3] Balog, K., Fang, Y., de Rijke, M., Serdyukov, P., Si, L.: Expertise retrieval. Foundations and Trends in Information Retrieval 6(2-3), 127–256 (2012)

[4] Brin, S.: Extracting patterns and relations from the world wide web. In: Atzeni, P., Mendelzon, A.O., Mecca, G. (eds.) WebDB 1998. LNCS, vol. 1590, pp. 172–183. Springer, Heidelberg (1999)

[5] Bron, M., Huurnink, B., de Rijke, M.: Linking archives using document enrichment and term selection. In: Gradmann, S., Borri, F., Meghini, C., Schuldt, H. (eds.) TPDL 2011. LNCS, vol. 6966, pp. 360–371. Springer, Heidelberg (2011)

[6] Chaudhari, D.L., Damani, O.P., Laxman, S.: Lexical co-occurrence, statistical significance, and word association. In: EMNLP 2011, pp. 1058–1068. ACL, Stroudsburg (2011)

[7] Elson, D.K., Dames, N., McKeown, K.R.: Extracting social networks from literary fiction. In: ACL 2010, pp. 138–147. ACL, Stroudsburg (2010)

[8] Etzioni, O., et al.: Open information extraction from the web. Commun. ACM 51(12), 68–74 (2008)

[9] Farkas, G.: Essays on Elite Networks in Sweden: Power, social integration, and informal contacts among political elites. PhD thesis, Stockholm University (2012)

[10] Finkel, J.R., Grenager, T., Manning, C.: Incorporating non-local information into information extraction systems by gibbs sampling. In: ACL 2005, pp. 363–370. Association for Computational Linguistics, Stroudsburg (2005)

[11] Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Mach. Learn. 63(1), 3–42 (2006)

[12] Harman, D.K., Voorhees, E.M. (eds.): TREC: Experiment and Evaluation in Information Retrieval. MIT Press (2005)

[13] Joachims, T.: Training linear SVMs in linear time. In: KDD 2006, pp. 217–226. ACM, New York (2006)

[14] Kautz, H., Selman, B., Shah, M.: Referral web: Combining social networks and collaborative filtering. Commun. ACM 40(3), 63–65 (1997)

[15] Klein, D., Manning, C.D.: Accurate unlexicalized parsing. In: ACL 2003, pp. 423–430. ACL, Stroudsburg (2003)

[16] Lunenfeld, P., Burdick, A., Drucker, J., Presner, T., Schnapp, J.: Digital Humanities. MIT Press (2012)

[17] Merhav, Y., Mesquita, F., Barbosa, D., Yee, W.G., Frieder, O.: Extracting information networks from the blogosphere. ACM Trans. Web 6(3), 11:1–11:33 (2012)

[18] Milne, D., Witten, I.H.: An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In: AAAI 2008 (2008)

[19] Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: ACL 2009, pp. 1003–1011. ACL, Stroudsburg (2009)

[20] Pedregosa, F., et al.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830 (2011)

[21] Tang, J., Zhang, D., Yao, L.: Social network extraction of academic researchers. In: ICDM 2007, pp. 292–301. IEEE Computer Society, Washington, DC (2007)

[22] Washtell, J., Markert, K.: A comparison of windowless and window-based computational association measures as predictors of syntagmatic human associations. In: EMNLP 2009, pp. 628–637. ACL, Stroudsburg (2009)