Workshop Programme

9:00 – 9:10 AM	Welcome Keith Miller, The MITRE Corporation
9:10 – 9:40 AM	*A Hybrid Method for Myanmar Named Entity Identification and Transliteration into English* Thi Thi Swe, Hla Hla Htay University of Computer Studies
9:40 – 10:00 AM	Introduction Keith Miller, The MITRE Corporation
10:00 – 10:30 AM	Group activity: name matching discussion and exercise
10:30 – 11:00 AM	Break
11:00 – 11:30 AM	*ENCORE: Experiments with a Synthetic Entity Co-reference Resolution Tool* Bo Lin, Rushin Shah, Robert Frederking, Anatole Gershman Carnegie Mellon University
11:30 – 12:00 PM	*Concord - A Tool that Automates the Construction of Record Resolution Systems* Christopher Dozier, Hugo Molina-Salgado, Merine Thomas, Sriharsha Veeramachaneni Thomson Reuters Research and Development
12:00 – 1:00 PM	Group activity: building an entity database from unstructured data
1 :00 – 1 :30 PM	*Identifcation, Extraction and Population of Collective Named Entities From Business News* Brett Drury, LIAAD-INESC J. J. Almeida, University of Minho
1:30 – 2:30 PM	Lunch
2:30 – 3:00 PM	*Name Matching Evaluation Framework* Dmitry Zelenko, SRA International
3:00 – 4:00 PM	Group discussion: current work in entity resolution
4:00 – 4:30 PM	Break
4:30 – 5:00 PM	*Resources for Named Entity Recognition and Resolution in News Wires* Rosa Stern, Alpage, Agence France-Presse Benoît Sagot, Agence France-Presse
5:00 – 5:30 PM	Group discussion: entity resolution resources
5:30 – 7:00 PM	Group discussion and wrap-up

Workshop Organisers

Keith J. Miller (The MITRE Corporation)

Mark Arehart (The MITRE Corporation)

Sherri Condon (The MITRE Corporation)

Chistopher Dozier (Thomson Reuters)

Jason Duncan (U.S. Department of Defense)

Louise Guthrie (University of Sheffield)

Emanuele Pianta (Fondazione Bruno Kessler)

Massimo Poesio (Universitá di Trento)

Elizabeth Schroeder (The MITRE Corporation)

Table of Contents

Author Index

A Hybrid Method for Myanmar Name Entity Extraction And Transliteration to English

Thi Thi Swe, Hla Hla Htay
University of Computer Studies
Yangon, Myanmar
E-mail: thithiswett@gmail.com, hlahlahtay123@gmail.com

Abstract

Named Entity Extraction (NEE) includes locating named entities and classifying those names in text. NEE is an important task in NLP applications such as Information Extraction, Cross Language information Retrieval, Question Answering, and Machine Translation. In this paper, we present a method for Myanmar Named Entity Identification using a hybrid method. This method is a combination of ruled based and statistical N-grams based method which use name databases. We have examined a sample of 43 Myanmar text files. The NEE experiment gives 82.75% precision, 83.40% recall on the sample data. We also classified those named entities into three classes. After that, those names are transliterated to their relevant Myanmar phonetics. We have used a transliteration table for mapping to English orthography. In the transliteration table, Myanmar syllables are transliterated into English pronunciation. 86.5% of recognized names are successfully transliterated to English and only 13.5% required phonetic adaptation. The system is implemented using Java.

1. Introduction

Named Entity Extraction (NEE) also known as Named Entity Recognition (NER) is a key technique in many text-based applications such as question answering, information retrieval, information extraction and machine translation.

There are two main problems in NEE. First, Named Entities (NE) are an open word class. Person, location, and organization names can be newly made by speakers of the language. Adding such words to dictionary is a very time consuming task and it is impossible to add all NEs to a dictionary. Second, NE can be used as several NE types (ambiguity problem). For example, “Paris” may be used as a person in the sentence “Paris was a prince of Troy.” and used as a location in the sentence “Paris is a capital city of France.” [8]

The English language has delimiters (for example, white space around words) and it has the significant definition of each word. Unlike English, Myanmar text is written with no space between words. Myanmar text does not also have case distinction. Therefore it is not trivial to find NEs in Myanmar language.

We present a hybrid method that combines two approaches, namely a Ruled Based Approach and a Statistical N-Grams Based Approach to identify the NEs. In the Ruled Based Approach, NEs are locating by using clue word lists, such as left occurrences, right occurrences and left-right co-occurrences. After locating NEs, those are classified according to their clue words. Most English NER systems use a statistics based approach, whereas this system uses both rules and statistics. Rule based NER systems can achieve proper performance with ease. In our system, rules are handcrafted.

Sometimes, NEs do not exist with clue words. In that situation, we perform NEE by using a Statistical N-Gram Based Approach. In second approach, three classes of names are collected and stored in the respective databases. Those names are syllabified [4] and unigram and bigram of those syllables are pre-calculated. To find name in a phrase, a phrase is firstly segmented into syllables [4]. And then, unigram and bigram of those syllables are calculated. Next, the frequencies of the syllables are calculated to identify whether that phrase is a NE or not. If the result is greater than zero, we assume that a phrase is a name. In this paper, we identify three kinds of Myanmar NEs, namely person name (PER), organization name (ORG) and location name (LOC).

2. Related Works

HuaPing Zhang et.al identified Chinese Named Entity using unified statistical model, namely role model. They defined roles as some special token class, including NE components and its neighboring and remote contexts. They also used the Viterbi algorithm to obtain tokens [1].

Hutchatai Chanlekha et.al reported on Thai Named Entity Recognition using statistical and heuristic rule-based models. The idea they used is to make use of a small proper name lexicon together with rules, created from internal and external evidence to extract Thai NEs [5].

Alireza Mansouri et.al presented English NER from text using FSVM for NER to improve the precision. They had employed Support Vector Machine for classification [2].

Fien De Meulder and Walter Daelemans described a memory-based approach to learning names in English and German newspaper text using the memory-based learner Timbl [3].

Richárd Farkas and György Szarvas described a multi lingual Named Entity Recognition (NER) system in

Hungarian texts that uses the statistical modeling techniques. They used the Support Vector Classifier, Artificial Neural Network and C4.5 decision tree learning algorithms. They focused on building as large a feature set as possible, and applied statistical preprocessing methods for feature selection afterwards to fully exploit its potential [7].

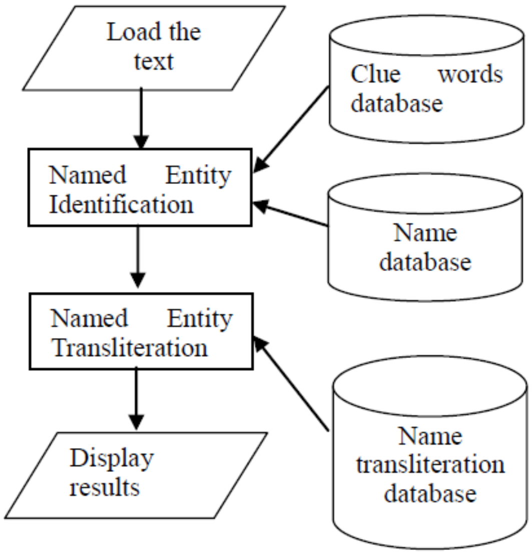
Yi-Gyu Hwang et.al presented a named entity recognition model for the Korean Language using an HMM-based named entity recognition using compound word construction principles [8].

3. Hybrid Method for Myanmar Named Entity Identification and Transliteration to English

In this system, we identify Myanmar Named Entity by using hybrid method that combines a rule-based approach and a statistical n-gram based approach and the resulting Nes are transliterated according to their relevant phonetics.

3.1 Rule Based Approach

In this approach, clue words are searched in the databases that are stored separately for respective NE types. The example clue words are seen in Table 1. A Named Entity can appear on the left hand side or right hand side of the clue words, and sometimes they may exist between clue words. Such cases will denote the left-right co-occurrence and may mostly occur in PERSON NEs.



Load the text
Clue words database
Named Entity Identification
Name database
Named Entity Transliteration
Name transliteration database
Display results

Figure 1 System Block Diagram

Roles	Examples	Numbers of Clue Words
Head components of Person or Neighboring token in front of NE	ဦး (Mr.) [u:], ဒေါ် (Mrs.) [do] , ကို (Mr.) [kou] , မောင် (Mr.) [maun] ,...	8
Tail components of Person or Neighboring token following NE	က (from) [ka], မှ (from) [hma], အား (to) [a:] , သည် (determiner) [the] ,...	22
Tokens between two NEs	နှင့် (and) [hnin.] , "၊" (comma) [pou' khalei:]	2
Tail component of organization or neighboring token following NE	ကုမ္ပဏီ (company) [koun pani] , ဌာနမှ (From Department) [hta na. hma] , ...	15
Tail component of location or neighboring token following NE	မြို့နယ် (township) [mjou ne], တိုင်း (division) [tain], ကျေးရွာ (kjei: jwa) [village], ဒေသ (region) [dei tha] , ပြည်နယ် (state) [pji ne],	80

Table 1: Role for three classes of NEs

Normally, person NEs can be found in the right hand side of the clue words. However, location and organization NEs can be found in the left hand side of the clue words. The phrase ဦးမြတ်မိုးအောင်အား (to U Myat Moe Aung) [u: mja' mou: aun a:] is taken as an example. In this case, if the phrase starts with clue word ဦး[1] and check whether the phrase ends with "က[2]" or "မှ[3]" or "အား[4]", etc. After that, the word between them is defined as a person NE. It is possible that a person name may start with left clue word but not end with a right clue word. The system could identify such case too.

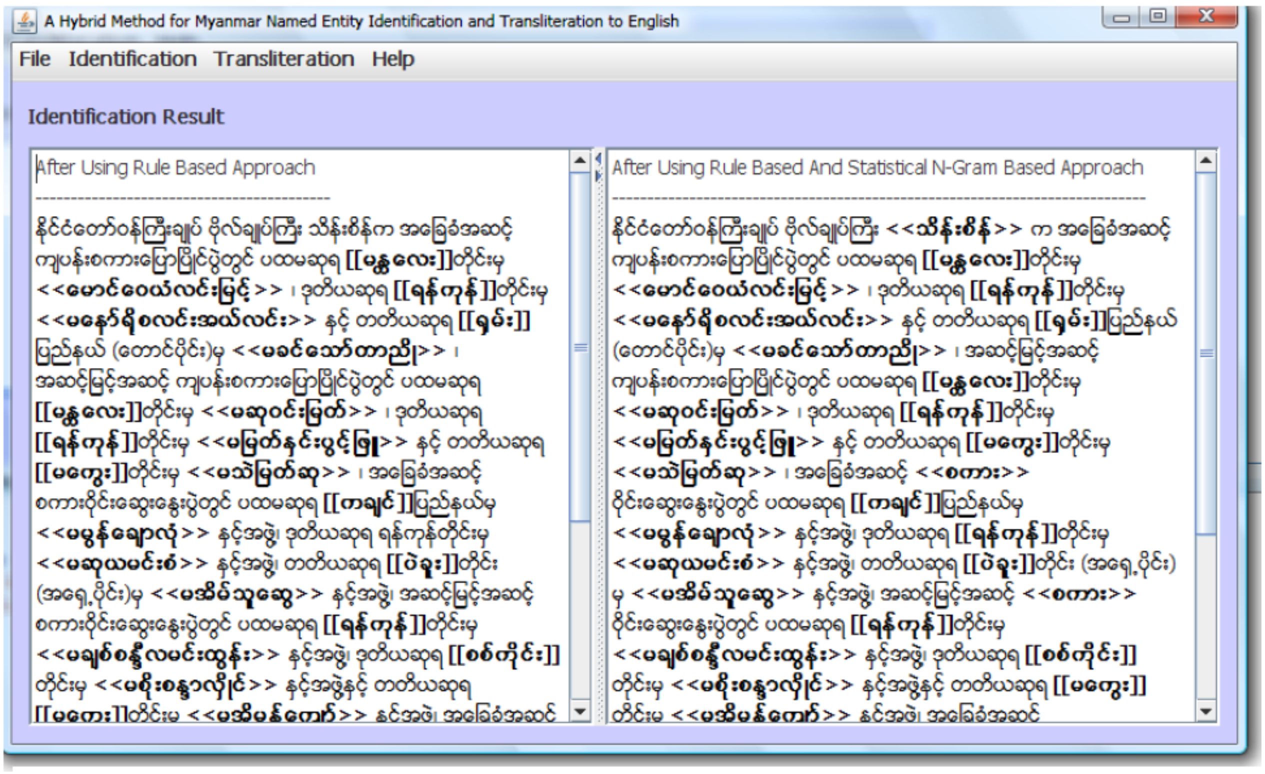
[1] Title prefixed to the name of a man
[2] Postpositional marker to indicate nominative case
[3] Postpositional marker equivalent to the locative proposition **from**
[4] Word indicating dative case equivalent to the preposition **to**

For example, ဦးမြတ်မိုးအောင် (U Myat Moe Aung) [u: mja' mou: aun] . For LOCATION and ORGANIZATION. NEs, there will be no clue right words. For example, ရန်ကုန်တိုင်း (Yangon Division) [jan koun tain:] , သုခစုစံကုမ္ပဏီ (Thu Kha Su San Company) [thu kha su. san koun pani] , etc. In ရန်ကုန်တိုင်း and သုခစုစံကုမ္ပဏီ, တိုင်း (Division) [tain:] and ကုမ္ပဏီ (Company) [koun pani] are clue words for location and organization. However, cases in which NEs occur without clue words will be solved with our statistically-based method.

3.2 Statistical N-Grams Based Approach

In this approach, we use the statistical n-grams based approach. Firstly, we collected over 10,000 person names and over 350 location names. Then, we pre-calculated the frequency of each unigram, bi-gram syllable by position likelihood, such as position$_1$, position$_2$, position$_{12}$, etc.

In this approach, we calculate the frequency of phrases that cannot be identified by our rule-based approach. Firstly, the phrase is syllabified and each syllable is treated as a gram. We then calculate the unidentified phrase's unigram, bi-gram syllable likelihood by position, and their frequencies are calculated by using our Name


A Hybrid Method for Myanmar Named Entity Identification and Transliteration to English
File
Identification
Transliteration
Help
Identification Result

After Using Rule Based Approach

နိုင်ငံတော်ဝန်ကြီးချုပ် ဗိုလ်ချုပ်ကြီး သိန်းစိန်က အခြေခံအဆင့်
ကျပန်းစကားပြောပြိုင်ပွဲတွင် ပထမဆုရ [[**မန္တလေး**]]တိုင်းမှ
<<**မောင်ဝေယံလင်းမြင့်**>> ၊ ဒုတိယဆုရ [[**ရန်ကုန်**]]တိုင်းမှ
<<**မနော်ရီစလင်းအယ်လင်း**>> နှင့် တတိယဆုရ [[**ရှမ်း**]]
ပြည်နယ် (တောင်ပိုင်း)မှ <<**မခင်သော်တာညို**>> ၊
အဆင့်မြင့်အဆင့် ကျပန်းစကားပြောပြိုင်ပွဲတွင် ပထမဆုရ
[[**မန္တလေး**]]တိုင်းမှ <<**မဆုဝင်းမြတ်**>> ၊ ဒုတိယဆုရ
[[**ရန်ကုန်**]]တိုင်းမှ <<**မမြတ်နှင်းပွင့်ဖြူ**>> နှင့် တတိယဆုရ
[[**မကွေး**]]တိုင်းမှ <<**မသဲမြတ်ဆု**>> ၊ အခြေခံအဆင့်
စကားဝိုင်းဆွေးနွေးပွဲတွင် ပထမဆုရ [[**ချင်း**]]ပြည်နယ်မှ
<<**မမွန်ချောလုံ**>> နှင့်အဖွဲ့၊ ဒုတိယဆုရ ရန်ကုန်တိုင်းမှ
<<**မဆုယမင်းစံ**>> နှင့်အဖွဲ့၊ တတိယဆုရ [[**ပဲခူး**]]တိုင်း
(အရှေ့ပိုင်း)မှ <<**မအိမ်သူဆွေ**>> နှင့်အဖွဲ့၊ အဆင့်မြင့်အဆင့်
စကားဝိုင်းဆွေးနွေးပွဲတွင် ပထမဆုရ [[**ရန်ကုန်**]]တိုင်းမှ
<<**မချစ်စန္ဒီလမင်းထွန်း**>> နှင့်အဖွဲ့၊ ဒုတိယဆုရ [[**စစ်ကိုင်း**]]
တိုင်းမှ <<**မစိုးစန္ဒာလှိုင်**>> နှင့်အဖွဲ့နှင့် တတိယဆုရ
[[**မကွေး**]]တိုင်းမှ <<**မအိမ်မန်ကော်**>> နှင့်အဖွဲ့၊ အခြေခံအဆင့်

After Using Rule Based And Statistical N-Gram Based Approach

နိုင်ငံတော်ဝန်ကြီးချုပ် ဗိုလ်ချုပ်ကြီး <<**သိန်းစိန်**>> က အခြေခံအဆင့်
ကျပန်းစကားပြောပြိုင်ပွဲတွင် ပထမဆုရ [[**မန္တလေး**]]တိုင်းမှ
<<**မောင်ဝေယံလင်းမြင့်**>> ၊ ဒုတိယဆုရ [[**ရန်ကုန်**]]တိုင်းမှ
<<**မနော်ရီစလင်းအယ်လင်း**>> နှင့် တတိယဆုရ [[**ရှမ်း**]]ပြည်နယ်
(တောင်ပိုင်း)မှ <<**မခင်သော်တာညို**>> ၊ အဆင့်မြင့်အဆင့်
ကျပန်းစကားပြောပြိုင်ပွဲတွင် ပထမဆုရ [[**မန္တလေး**]]တိုင်းမှ
<<**မဆုဝင်းမြတ်**>> ၊ ဒုတိယဆုရ [[**ရန်ကုန်**]]တိုင်းမှ
<<**မမြတ်နှင်းပွင့်ဖြူ**>> နှင့် တတိယဆုရ [[**မကွေး**]]တိုင်းမှ
<<**မသဲမြတ်ဆု**>> ၊ အခြေခံအဆင့် <<**စကား**>>
ဝိုင်းဆွေးနွေးပွဲတွင် ပထမဆုရ [[**ချင်း**]]ပြည်နယ်မှ
<<**မမွန်ချောလုံ**>> နှင့်အဖွဲ့၊ ဒုတိယဆုရ [[**ရန်ကုန်**]]တိုင်းမှ
<<**မဆုယမင်းစံ**>> နှင့်အဖွဲ့၊ တတိယဆုရ [[**ပဲခူး**]]တိုင်း (အရှေ့ပိုင်း)
မှ <<**မအိမ်သူဆွေ**>> နှင့်အဖွဲ့၊ အဆင့်မြင့်အဆင့် <<**စကား**>>
ဝိုင်းဆွေးနွေးပွဲတွင် ပထမဆုရ [[**ရန်ကုန်**]]တိုင်းမှ
<<**မချစ်စန္ဒီလမင်းထွန်း**>> နှင့်အဖွဲ့၊ ဒုတိယဆုရ [[**စစ်ကိုင်း**]]
တိုင်းမှ <<**မစိုးစန္ဒာလှိုင်**>> နှင့်အဖွဲ့နှင့် တတိယဆုရ [[**မကွေး**]]
တိုင်းမှ <<**မအိမ်မန်ကော်**>> နှင့်အဖွဲ့၊ အခြေခံအဆင့်

Figure 2 Named Entities identified Myanmar texts

```
For each phrase in the sentence do
  ArraySyllables ← Syllabification(Phrase)
  length ← ArraySyllables
  While (length>2) do
    For (i=1..length-1)
      $\text{result} = \prod_{i=1}^{i=m-1} \left[ \frac{\text{freq}(\text{pos}_i(X_i))}{\text{freq}(X_i)} * \frac{\text{freq}(\text{pos}_i(X_i), \text{pos}_{i+1}(X_{i+1}))}{(\text{freq}(\text{pos}_i(X_i)) + \text{freq}(\text{pos}_{i+1}(X_{i+1})))} \right] \frac{\text{freq}(\text{pos}(X_m))}{\text{freq}(X_m)}$
    End for
    If (result==0)
      length ← length-1
    Else
      Identify that phrase as a NE
      NE found. Match NE.
      Break
    End if
  End while
End for
```

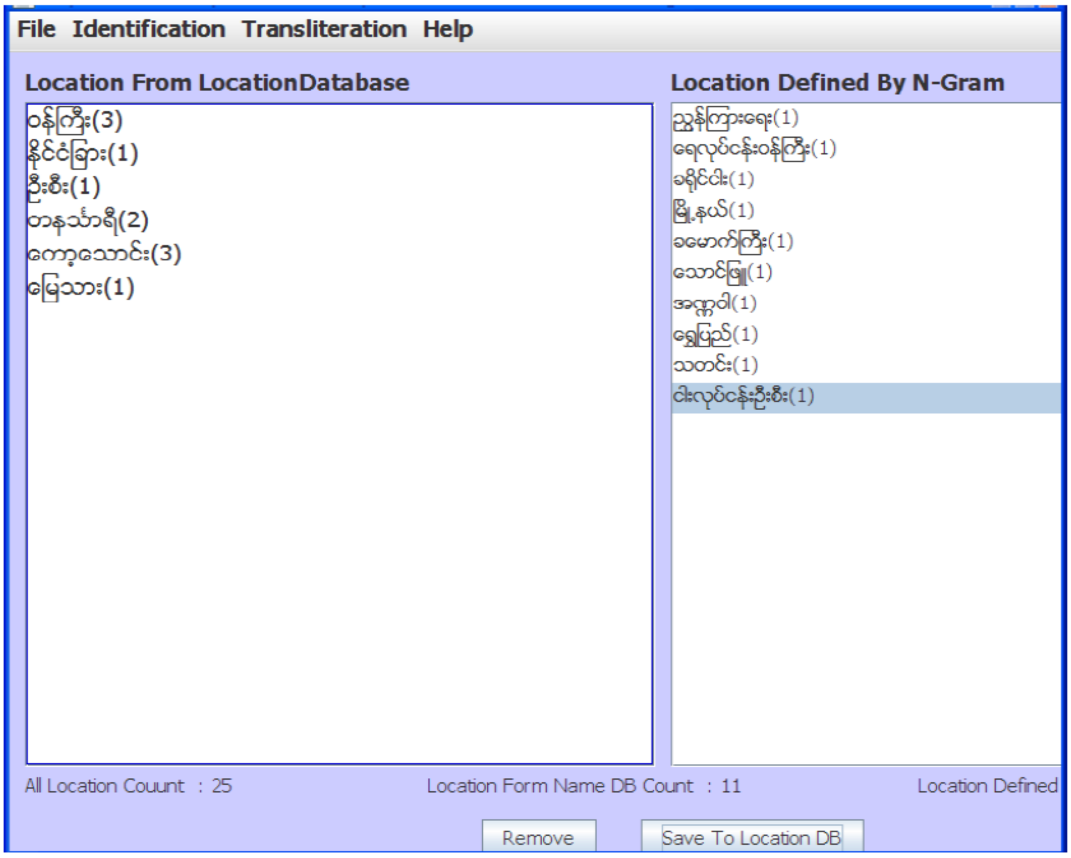
and Location database. For example, in the phrase နွယ်နီချိုအား (to Nwe Ni Cho) [nwe ni chou a:] , position$_1$ is နွယ် , position$_2$ is နီ and position$_{12}$ is နွယ်နီ . In which [နွယ်နီချိုအား], နွယ်|နီ|ချို is NEs and အား is just a particle follow by a NE. In identifying နွယ်နီချို as a NE, two iterations are required as the နွယ်|နီ|ချို|အား is a 4-syllabled phrase. .The equation [6] is iteratively calculated the N-gram frequencies in the following pseudo code.

In first iteration, နွယ်နီချိုအား is calculated as followed.

$$= \left[\left(\frac{\mathbf{freq}(\mathbf{pos}_1(\text{နွယ်}))}{\mathbf{freq}(\text{နွယ်})} * \frac{\mathbf{freq}(\mathbf{pos}_1\mathbf{pos}_2(\text{နွယ်နီ}))}{(\mathbf{freq}(\mathbf{pos}_1(\text{နွယ်}))) + \mathbf{freq}(\mathbf{pos}_2(\text{နီ}))} \right) * \left(\frac{\mathbf{freq}(\mathbf{pos}_2(\text{နီ}))}{\mathbf{freq}(\text{နီ})} * \frac{\mathbf{freq}(\mathbf{pos}_2\mathbf{pos}_3(\text{နီချို}))}{(\mathbf{freq}(\mathbf{pos}_2(\text{နီ}))) + \mathbf{freq}(\mathbf{pos}_3(\text{ချို}))} \right) * \left(\frac{\mathbf{freq}(\mathbf{pos}_3(\text{ချို}))}{\mathbf{freq}(\text{ချို})} * \frac{\mathbf{freq}(\mathbf{pos}_3\mathbf{pos}_4(\text{ချိုအား}))}{(\mathbf{freq}(\mathbf{pos}_3(\text{ချို}))) + \mathbf{freq}(\mathbf{pos}_4(\text{အား}))} \right) \right] \left(\frac{\mathbf{freq}(\mathbf{pos}_4(\text{အား}))}{\mathbf{freq}(\text{အား})} \right)$$

$$= \left[\left(\frac{\mathbf{freq}(\mathbf{pos}_1(\text{နွယ်}))}{\mathbf{freq}(\text{နွယ်})} * \frac{\mathbf{freq}(\mathbf{pos}_1\mathbf{pos}_2(\text{နွယ်နီ}))}{(\mathbf{freq}(\mathbf{pos}_1(\text{နွယ်}))) + \mathbf{freq}(\mathbf{pos}_2(\text{နီ}))} \right) * \left(\frac{\mathbf{freq}(\mathbf{pos}_2(\text{နီ}))}{\mathbf{freq}(\text{နီ})} * \frac{\mathbf{freq}(\mathbf{pos}_2\mathbf{pos}_3(\text{နီချို}))}{(\mathbf{freq}(\mathbf{pos}_2(\text{နီ}))) + \mathbf{freq}(\mathbf{pos}_3(\text{ချို}))} \right) \right] \left(\frac{\mathbf{freq}(\mathbf{pos}_3(\text{ချို}))}{\mathbf{freq}(\text{ချို})} \right)$$

This approach is time consuming if the identified text is long. Mostly, Myanmar NEs do not usually start with clue words. Therefore, the second approach is very efficient and useful in identification of NEs that do not start with clue words. The recognized names can be stored in the respective databases for future use. But the process must be done with the help of the human judgment. See details in figure 3.



Location From LocationDatabase

ဝန်ကြီး(3)
နိုင်ငံခြား(1)
ဦးစီး(1)
တနင်္သာရီ(2)
ကော့သောင်း(3)
မြေသား(1)

Location Defined By N-Gram

ညွှန်ကြားရေး(1)
ရေလုပ်ငန်းဝန်ကြီး(1)
ခရိုင်ငါး(1)
မြို့နယ်(1)
ခမောက်ကြီး(1)
သောင်ဖြူ(1)
အဏ္ဏဝါ(1)
ရွှေပြည်(1)
သတင်း(1)
ငါးလုပ်ငန်းဦးစီး(1)

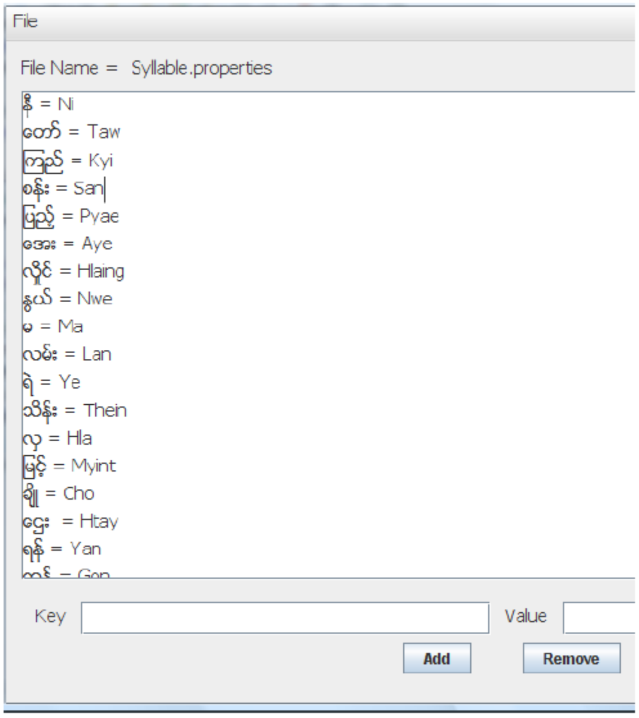
All Location Count : 25 Location Form Name DB Count : 11 Location Defined

Figure 2 Location Name Manager

In above calculation, the frequency result of နွယ်နီချိုအား is zero because there is no unigram syllable အား in name database and the frequency value of အား is zero. Therefore, နွယ်နီချိုအား cannot be identified as a NE. In *second iteration*, နွယ်နီချို is calculated again decreasing length 1. As frequency result is greater than zero, that phrase is identified as a NE.

In figure 3, the left hand side displays the location name recognized by the location database and the right hand side displays the location name recognized by using N-Gram statistical based approach. Those recognized names can be removed or added to the location database for future.

4. Transliteration of Identified Name Entities to English



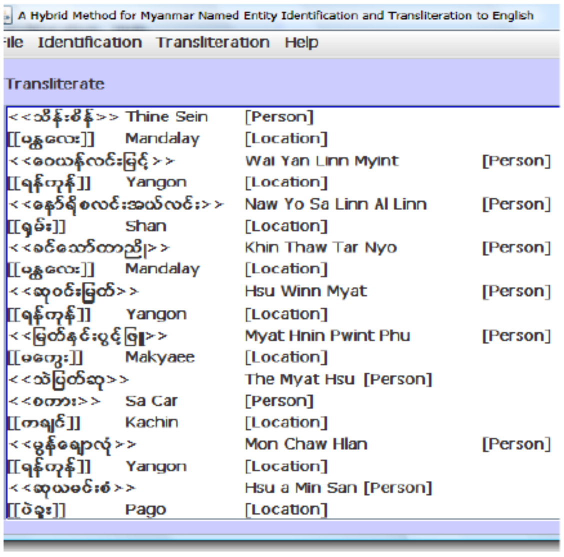
File

File Name = Syllable.properties

```
နီ = Ni
တော် = Taw
ကြည် = Kyi
စန်း = San
ပြည့် = Pyae
အေး = Aye
လှိုင် = Hlaing
နွယ် = Nwe
မ = Ma
လမ်း = Lan
ရဲ = Ye
သိန်း = Thein
လှ = Hla
မြင့် = Myint
ချို = Cho
ဌေး = Htay
ရန် = Yan
```

Figure 3 Transliteration Table

Transliteration is the process of replacing words in a source script with their approximate phonetic or spelling equivalents in a target script. Commonly, transliteration is used to translate named entities across languages. Automatic Transliteration is helpful for many applications such as Machine Translation, Cross Language Information Retrieval and Information Extraction.



A Hybrid Method for Myanmar Named Entity Identification and Transliteration to English
File Identification Transliteration Help
Transliterate

```
<<သိန်းစိန်>> Thine Sein        [Person]
[[မန္တလေး]]    Mandalay         [Location]
<<ဝေယံလင်းမြင့်>>               Wai Yan Linn Myint          [Person]
[[ရန်ကုန်]]    Yangon           [Location]
<<နော်ရိုစလင်းအယ်လင်း>>          Naw Yo Sa Linn Al Linn      [Person]
[[ရှမ်း]]      Shan             [Location]
<<ခင်သော်တာညို>>               Khin Thaw Tar Nyo           [Person]
[[မန္တလေး]]    Mandalay         [Location]
<<ဆုဝင်းမြတ်>>                  Hsu Winn Myat               [Person]
[[ရန်ကုန်]]    Yangon           [Location]
<<မြတ်နှင်းပွင့်ဖြူ>>              Myat Hnin Pwint Phu         [Person]
[[မကွေး]]     Makyaee          [Location]
<<သဲမြတ်ဆု>>                    The Myat Hsu [Person]
<<စကား>>      Sa Car           [Person]
[[ကချင်]]     Kachin           [Location]
<<မွန်ချောလှံ>>                  Mon Chaw Hlan               [Person]
[[ရန်ကုန်]]    Yangon           [Location]
<<ဆုယမင်းစံ>>                   Hsu a Min San [Person]
[[ပဲခူး]]      Pago             [Location]
```

Figure 4 Transliterated Named Entities

For transliteration of Myanmar names to English, Myanmar syllables and their associated English transliterations are stored in an equivalence table The entries can be seen in Figure 3. Currently, the file contains 4096 entries. For example, the recognized person name နွယ်နီချို is syllabified and it has three syllables. Each syllable is looked up in the that **table**. After looking up all the three syllables, the resultant transliteration for နွယ်နီချို is "Nwe Ni Cho". Figure 4 shows transliterated name entities in the input text. In the transliteration, location names are written without spaces between syllables and persons are usually written with space between syllables.

5. Evaluation and Error Analyses

The sample text contains news data from မြန်မာ့အလင်း (New Light of Myanmar) [mjanma. alin] newspaper (Nov 2009 - March 2010) which are available in electronic version in Zawgyi font script in http://www.usda.org.mm. The Zawgyi font is semi-unicode and popular among internet users. The sample data contains 178 sentences (43 files/paragraphs). It includes 388 PER, 288 LOC, and 37 ORG. We conduct evaluations in terms of precision, recall, and F-measure. The result can be seen in Table 2.

$$Precision = \frac{Number\ of\ correctly\ identified\ NEs}{Number\ of\ identified\ NEs}$$

$$Recall = \frac{Number\ of\ corretly\ identifed\ NEs}{Number\ of\ all\ NEs}$$

$$F - measure = \frac{2 * P * R}{P + R}$$

	PER	LOC	ORG	Total
Precision	78.97	88.07	77.08	82.75
Recall	79.38	89.16	77.08	83.40
F-measure	79.18	88.62	77.08	83.07

Table 2 Result on the sample data

We have also examined the errors in all three stages of our system.

1. **Named Entity Identification**: The rule-based system wrongly identified as named entities some phrase starting or ending with a clue word and in fact, the phrase is open class word. For examples, ဦးဦးဖျားဖျား (first and foremost) [u: u: hpja: bja:], မနှစ်မြို့ (donot like) [ma hni' mjou.] But this error can be corrected by checking statistical n-grams based approach.
2. **Classification:** Sometimes a part of a phrase was misidentified as named entity. For example စုစုပေါင်း (total) [su. zu. baun:] where the first two syllables are frequently used for ladies' name. This error can be corrected if the dictionary is used.
3. **Transliteration**: The identified names are needed to

improve the phonetic adaptation according to the syllable position although adaptation is required 13.5% of the names found. For example, the location named ပုဇွန်တောင် will be wrongly transliterated into "Pu|Zun|Taung" However, in this name, the first and last syllables are needed phonetic adaptation. Therefore, the correct transliteration is "Pa|Zun|Daung". The phonetic adaptation errors can be solved with machine learning techniques.

6. Conclusion

Named Entity Identification includes locating name entities and classifying those names in text. NEI is important in NLP applications such as Cross Language Information Retrieval, Question Answering, and Machine Translation. NEI is important in NLP applications such as Information Extraction, Cross Language Information Retrieval.

In this paper, we presented a method for Myanmar Named Entity Identification using a hybrid method which was a combination of rule-based and statistical N-grams based methods. We have conducted evaluation for NE extraction on a sample data of 43 Myanmar text files/paragraphs. It contains 178 sentences and includes 388 PER, 288 LOC, and 37 ORG. The precision of PER, LOC, ORG on the sample data is 78.97%, 88.07%, 77.08% respectively; and the recall is 79.38%,89.16%,77.08% respectively.

After that, the identified names are transliterated to their relevant Myanmar phonetics. We have used a transliteration table for mapping to English orthography. In transliteration table, Myanmar syllables are transliterated into Roman script. 86.5% of recognized names are successfully transliterated to English and only 13.5% required the phonetic adaptation. The system is implemented using Java. We have also discussed the errors and suggested how to tackle those errors. The system can also combine in Myanmar word segmentation process and Myanmar part of speech tagging. All databases can be updated.

7. Acknowledgment

Special thanks go to reviewer Keith J. Miller who provided extremely detailed comments and many useful suggestions for improving this paper.

8. References

Huaping Zhang, Qun Liu, Hongkui Yu, Xueqi Cheng, Shuo Bai: Chinese Named Entity Recognition Using Role Model. In: the International Journal of Computational Linguistics and Chinese Language Processing, vol.8, No.2. (2003) 29-60.

Alireza Mansouri, Lilly Suriani Affendey, Ali Mamat, "Named Entity Recognition Using a New Fuzzy Support Vector Machine", IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.2, 320-325, February 2008.

Fien De Meulder, Walter Daelemans, "Memory-Based Named Entity Recognition using Unannotated Data", Human Language Technology Conference, In Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 ,Vol 4, 208-211, Edmonton, Canada.

Hla Hla Htay and Kavi Narayana Murthy, Myanmar Word Segmentation using Syllable level Longest Matching, The 6thWorkshop on Asian Language Resources (ALR 6) , January 2008, Hyderabad, India.

Hutchatai Chanlekha, Asanee Kawtrakul, Patcharee Varasrai and Intiraporn Mulasas, " Statistical and Ruled Based Model for Thai Named Entity Recognition", In Proceedings of Joint International Conference of SNLPOriental COCOSDA 2002, 326-329.

Paul Wu Horng-Jyh, Na Jin-Cheon, Christopher Khoo Soo-Guan , "A Hybrid approach to fuzzy name search incorporating language-based and text based principles", Journal of Information Science, , 33(1), 2007, pp. 3-19.

Richárd Farkas, György Szarvas, "Statistical Named Entity Recognition for Hungarian – analysis of the impact of feature space characteristics", in Proceedings of CESCL 2006 Budapest, Hungary.

Yi-Gyu Hwang, Eui-Sok Chung, Soo-jong Lim, "HMM based Korean Named Entity Recognition", Journal of the Korean Information Processing Society(B), vol.10, No.2, pp. 229-236, 2003.

Myanmar-English Dictionary, Department of the Myanmar Language Commission, Ministry of Education, Myanmar, 1993.

Jian Sun, Jianfeng Gao, Lei Zhang, Ming Zhou, Changning Huan, "Chinese named entity identification using class-based language model", in Proceedings of the 19th International Conference on Computational Linguistics (COLING-2002), Taipei, 2002, 967-973.

ENCORE: Experiments with a Synthetic Entity Co-reference Resolution Tool

Bo Lin, Rushin Shah, Robert Frederking, Anatole Gershman
Language Technologies Institute, School of Computer Science, Carnegie Mellon University
5000 Forbes Ave., Pittsburgh 15213, PA, USA
{bolin, rnshah, ref+, anatoleg}@cs.cmu.edu

Abstract

We present ENCORE, a system for entity co-reference resolution that synthesizes the outputs of several off-the-shelf co-reference resolution systems. To boost precision, we filter the output using a named entity recognition tool called SYNERGY which itself is a synthesis of several off-the-shelf NER systems. ENCORE is designed to work under two conditions: NP-CR which resolves noun phrase co-reference and NE-CR which resolves co-references only for named entities. We report the results of our experiments with ENCORE that show 2% to 40% improvements in precision, recall and F-scores over the underlying systems. This opens a promising approach which leverages the existing "black box" state-of-the-art tools without attempting to re-create their achievements and focuses the development efforts on the differences in their output.

1. Introduction

"Co-reference resolution is the process of determining whether two expressions in natural language refer to the same entity in the world" (Soon et al., 2001). It is a critical task in information extraction and it has received much attention in the last decade using both rule-based and machine learning approaches. As a result, there is a growing number of proprietary and open-source co-reference resolution systems (Versley et al., 2008; Bengston & Roth, 2008). Their performance is typically in the 50% to 70% F-measure range on various metrics, which leaves substantial room for improvement. More importantly, many tools tend to specialize in particular areas such as foreign names or biological entities. It would be virtually impossible to re-create all of their best features in a single tool. A contrasting approach would be to create a system capable of using a variety of available tools as black boxes, leveraging their individual capabilities, integrating and improving their collective results. This approach is challenging because the underlying tools were not created for the purpose of integration. The tools also change as their authors introduce new features and improve performance.

In this paper, we propose a novel synthetic tool called ENCORE that provides superior performance by leveraging several of the best state-of-the-art tools. We treat the underlying tools as co-reference annotators and developed several heuristics that examine their results and create synthetic co-reference classes. Our tests show 2% to 27% and 20% to 40% improvements under two different conditions over the underlying systems in precision, recall and F-scores on two test sets. Section 2 of this paper includes a brief review of related work, Section 3 describes our evaluation metrics, Section 4 introduces our integration methodology, and Section 5 provides the results of our experiments.

Finally, a word on terminology: we use the terms "textual reference," "reference" and "mention" interchangeably to refer to a text phrase. We use the terms "entity," "object" and "equivalence class" to refer to real-world objects.

2. Related Work

Current state-of-the-art approaches include both rule-based and machine learning algorithms. The rule-based approaches apply inductive logic programming, which combines rules for co-reference resolution in a logic induction framework. Other researchers use Markov logic networks with a probabilistic version of logic induction (Culotta et al., 2007). Many researchers have explored machine learning approaches by treating the problem as a pair-wise binary classification problem with subsequent entity clustering or a joint model of classification and clustering (Soon et al., 2001; Ng & Cardie, 2002; Ng, 2005; Haghighi & Klein 2007; Ng, 2008; Finkel & Manning, 2008). The most recent work (Haghighi & Klein, 2009) focuses on feature analysis with a simple model for co-reference resolution. With its rich set of syntactic and semantic features, it is reported to outperform the current state-of-the-art systems.

In the work reported in this paper, we focus on the leveraging of publicly available tools for co-reference resolution. One of them is from a recent study on the value of using rich features for co-reference resolution (Bengtson & Roth, 2008) which comes as a Learning-Based Java (LBJ) co-reference package. Another is called BART, which is from the Johns Hopkins University summer workshop on using lexical and encyclopedic knowledge for entity disambiguation (Versley et al., 2008). We used both of these tools as described in Section 4.

3. Evaluation Metrics

Evaluation for co-reference resolution is challenging, considering that co-reference resolution is neither a traditional classification problem nor a labelling problem. A good evaluation metric has to consider both entity recognition and clustering. Several efforts have been made to establish standard evaluation metrics. Link-based F-measure is the one of the earliest metrics adopted in the MUC task (Vilain et al., 1995). In this metric the F-measure is computed on the co-reference links from the system output against the links in the gold standard. However, it is reported to be biased for systems with fewer entity outputs (Finkel & Manning, 2008; Luo, 2005). Another metric, called B-cubed (B^3) takes the weighted sum of the F-measures for each individual mention which helps alleviate the bias in the pure link-based F-measure (Bagga & Baldwin, 1998). The ACE named entity detection and tracking task used a metric that normalizes the sum of false-alarm, missed and mistaken entities. However, this metric is believed to be non-intuitive and hard to interpret (Luo, 2005). Instead Luo proposed a new metric named Constrained Entity Alignment F-measure (CEAF), which is claimed to be both discriminative and interpretable. CEAF uses a set similarity measure $\Phi(A, B)$ that is simply the number of common elements in sets A and B. CEAF compares the equivalence classes R produced by the system to the classes in the gold standard G. It calculates the mapping $g{:}R \rightarrow G$ that maximizes the sum of all $\Phi(R_i, g(R_i))$. The optimal mapping is used to define precision, recall and F-measure as follows:

$$p = \frac{\sum \Phi(R_i, g(R_i))}{\sum |R_i|}$$

$$r = \frac{\sum \Phi(R_i, g(R_i))}{\sum |G_i|}$$

$$f = \frac{2pr}{p + r}$$

In this paper, we present our results using two widely used metrics, B^3 and CEAF, and show our improvement over the baselines.

In addition to the selection of the appropriate metrics, we address the issue of the target entities selection for the co-reference resolution task. Typically, the Entity Co-reference Resolution (ECR) systems try to extract all potential references to real-world objects, including both named and nameless objects. For example, the phrase "a Canadian company" is considered a reference, albeit to a nameless object. The documents in the ACE collection are annotated according to this convention. In our conversations with many potential users of the ECR systems we noted that their major interest was in named objects only. For example, a reference to "a Canadian company" would not be interesting unless that company could be identified by name elsewhere the text. Correspondingly, we established two experimental conditions: the traditional condition of NP-CR (Noun Phrase Co-reference Resolution) and NE-CR (Named Entities Co-reference Resolution). For NE-CR, we manually re-annotated the original gold standard to include only NE-CR references.

4. ENCORE Details

In this section, we present the details of our synthetic co-reference resolution system. The system uses several off-the-shelf co-reference resolution tools and integrates their results. We first discuss these state-of-the-art co-reference resolution tools and then propose our own integration methodology.

4.1 State-of-the-art Tools

For our experiments we selected four popular, publicly available ECR packages: (1) the Learning-Based Java ("LBJ System") co-reference package from UIUC (Bengtson & Roth, 2008), (2) BART co-reference toolkit from Johns Hopkins University Summer Workshop (Versley et al., 2008) and two others. In our testing, LBJ and BART produced results uniformly superior to the other two systems with very little new information provided by the latter two. For this reason, we used only LBJ and BART in ENCORE. LBJ and BART embody different approaches to co-reference resolution. While LBJ incorporates a rich set of syntactic and semantic features, BART additionally uses information from Wikipedia for co-reference disambiguation (Versley et al., 2008). Our expectation is that the different approaches adopted by LBJ and BART would produce complementary results with room for further refinement. We conducted a pilot test on a manually annotated corpus consisting of 10 articles from politics, sports and entertainment with about 500 entities and 100 equivalence classes. The pilot showed that the two systems are rather complementary because a simple union of the co-reference chains discovered by each system resulted in a 10% increase in recall. The pilot also gave us ideas on possible heuristics which we will detail in the following sections. On other sets (see below), we achieved an even greater increase in recall. This number indicates an upper bound for the integration heuristics if they can prevent a deterioration of precision caused by false positives.

4.2 Integration Methodology

In this section, we present our initial integration heuristics for combining the outputs of the primary co-reference resolution tools. We treat each primary system as an annotator that marks text phrases with the labels of the corresponding entities (objects). Each entity produced by the annotator has a set of textual references – some that contain names and some that do not.

First, ENCORE tries to merge entities produced by different annotators. The first heuristic is rather obvious: if two such entities have identical lists of textual references, they are merged. The second heuristic is more interesting and important. We merge two entities if their lists of textual references have at least one *pivot* reference in common. Which reference qualifies as a pivot reference is of critical importance. Allowing all references to serve as pivots, often leads to serious mistakes. For example, a common pronoun could merge two entities which may not be the same, as in:

(John$_1$, he$_2$) and (he$_2$, Bill$_3$)

On the other hand, if the selection criteria for pivot references are too restrictive then we may end up with too many entities and many ambiguous references like "he$_2$" in the above example, attached to two different entities.

ENCORE uses different pivot criteria in our two experimental conditions. Under the NE-CR condition, only named textual references are used as pivots. Our preliminary investigation indicated that the nominal (nameless) textual references where there is no agreement between the primary co-reference resolution tools are most likely to cause errors if we use them as pivots. Their elimination improves precision without significant reduction of recall. To identify named references we use SYNERGY, our own NER (Named Entity Extractor) (Shah et al., 2010) which synthesizes the results from several off-the-shelf NER systems, but any other high-performance NER system could be used as well.

Under the NP-CR condition, the above restriction on pivots turned out to be too narrow, leaving out too many valid references, especially the classes consisting of only the nominal textual references. For this condition we developed a more "relaxed" version, counting as pivots all phrases that either contain a named reference or are contained in one. For example, the phrase "the President of the United States" is not a named reference (the president is not identified) but it contains a named reference "the United States" which makes it a pivot reference. The textual reference "Properties," while not a named reference, can serve as a pivot if it is contained in the phrase "Hong Kong Properties LTD". In our experiments, mentions "associated" with named references in the above manner were more reliable as pivots, increasing the accuracy of merging.

After the mergers, each named reference belongs to one and only one entity. Nameless references might belong to several entities. ENCORE cleans up the results using the following heuristics. Nameless references that belong to more than one entity are eliminated as ambiguous. Under the NE-CR condition, entities (sets of references) that contain no named references are eliminated.

The following example illustrates how ENCORE works in general under both conditions:

LBJ produces the following two equivalence classes (among many others):

(J. Smith, Joe Smith)

(an oil company)

BART system produces only one class containing

(J. Smith, Mr. Smith)

The reference "J. Smith" is recognized by SYNERGY as a named entity, and the first two classes are merged, producing:

(J. Smith, Joe Smith, Mr. Smith)

The reference "an oil company" is not recognized by SYNERGY and is dropped.

The following example illustrates under NE-CR, how ENCORE is able to improve on the underlying systems without getting confused by a significant error in one of them:

Our friend **Jakaya Kikwete**$_1$ studied at that school last summer.... By the way, **he**$_2$ is marrying **Maria Kashonda**$_3$ soon.... **Jakaya**$_4$ and **Maria**$_5$ are moving to Teheran where **he**$_6$'ll be working for Pishgaman Nano Arya.

LBJ produces three classes where it misclassifies the last "he" as a reference to "Maria" which is clearly a female name.

LBJ1 = (Jakaya Kikwete$_1$, Jakaya$_4$)

LBJ2 = (Maria Kashonda$_3$, Maria$_5$, he$_6$)

LBJ3 = (he$_2$)

BART returns the following classes:

BART1 = (Jakaya Kikwete$_1$, he$_2$, Jakaya$_4$, he$_6$)

BART2 = (Maria$_5$)

ENCORE merges LBJ1 with BART1 and LBJ2 with BART2. It does not merge LBJ2 and BART1 because "he$_6$" is not a pivot reference. It then eliminates LBJ3 and "he$_6$" producing:

ENT1 = (Jakaya Kikwete$_1$, he$_2$, Jakaya$_4$)

ENT2 = (Maria Kashonda$_3$, Maria$_5$)

In the process, we lost the reference he$_6$ which should have been part of ENT1, but was eliminated because of its ambiguity. A more sophisticated heuristic aware of first name genders would have salvaged it.

5. Experiments

We conducted our experiments on two test sets: MIX1 and ACE NWIRE. MIX1 is a small set of 10 articles that reflect one of our target application domains: business news and biographical sketches. This manually annotated set contains approximately 4000 words, 500 entities and 100 co-reference equivalence classes. The ACE NWIRE set from ACE-2 corpus for NIST Automatic Content Extraction program is widely used in co-reference resolution experiments (Mitchell et al., 2003). It is based on 29 articles and contains approximately 20,000 words, 2600 entities and 1000 co-reference equivalence classes. MIX1 set was too small for meaningful tests under the NE-CR condition. To conduct the tests, we manually created the annotations of the ACE NWIRE corpus with equivalence classes consisting of only named entities and their references (both named and nameless).

Some earlier systems (Bengston & Roth, 2008; Haghighi & Klein 2009) achieved good results, but they matched only head nouns between entities in the gold standard and those in the system output, instead of the entire mentions.

We should note that using entire mentions instead of just head nouns makes evaluation much stricter. We follow this approach, for two reasons: Firstly, our main objective is to show the improvement over baseline systems. By placing strict rules of evaluation, it would be more effective to observe the direct improvements from ENCORE. Secondly, the underlying primary systems, as black boxes, return only the full extends of textual mentions/references with no head noun information for our evaluation.

	Precision	Recall	F-Score
ENCORE	**0.468**	**0.462**	**0.458**
Union	0.256	**0.530**	0.340
LBJ	0.342	0.479	**0.396**
BART	0.404	0.400	**0.387**

Table 1: MIX1 with CEAF under NP-CR

	Precision	Recall	F-Score
ENCORE	**0.410**	**0.425**	**0.395**
Union	0.220	**0.548**	0.301
LBJ	0.346	0.377	**0.356**
BART	0.352	0.354	**0.333**

Table 2: MIX1 with B^3 under NP-CR

For comparison, we used three baselines: the individual performances of LBJ and BART, and another baseline named Union created by us. Union measures the results of a simple union of the equivalence classes retrieved by the two underlying systems. Union should give us the maximum recall achievable by the integration of the underlying systems at the cost of diminished precision. The results of our experiments are shown in Table 1 and Table 2 for the MIX1 test set under the NP-CR condition, Table 3 and Table 4 for the ACE NWIRE test set also under the NP-CR condition, and Table 5 and Table 6 for the ACE NWIRE test set under the NE-CR condition.

5.1 Tests under NP-CR Condition

As we can see from these tables, ENCORE produces better precision, better F-score and comparable recall results as compared to the two underlying systems. On the MIX1 test set under NP-CR condition, ENCORE shows 16% and 18% improvement with CEAF over the individual F-scores of LBJ and BART. Similar improvements of 19% and 27% are observed with B^3. However, the recall results of ENCORE are significantly below the maximum, as indicated by the Union baseline. The 0.462 recall value of ENCORE is about 13% below the recall value of the Union baseline with CEAF. With B^3 the difference is 23%. Our current heuristics reject many valid nominal (nameless) mentions. This leaves ample headroom for future improvements with better heuristics and the use of machine learning techniques for synthesis.

	Precision	Recall	F-Score
ENCORE	**0.525**	**0.521**	**0.512**
Union	0.417	**0.546**	0.465
LBJ	0.485	0.505	**0.493**
BART	0.441	0.386	**0.407**

Table 3: ACE NWIRE with CEAF under NP-CR

	Precision	Recall	F-Score
ENCORE	**0.511**	**0.476**	**0.481**
Union	0.393	**0.536**	0.441
LBJ	0.533	0.446	**0.476**
BART	0.391	0.320	**0.340**

Table 4: ACE NWIRE with B^3 under NP-CR

As shown in the F-Score column of Table 3 and 4, ENCORE improves the F-scores on the ACE NWIRE by 1.7% and 24.8% over LBJ and BART respectively with CEAF. It also gives improvements of 2.7% and 29.0% over the two baselines with B^3.

The improvements on the ACE NWIRE test set is considerably lower than that on MIX1 test set. Close examination of the results on the individual files of the ACE NWIRE set reveals that on this test set, LBJ consistently outperforms BART for most of the files. Yet the latter system still contributes enough differences to improve the precision score over the LBJ system by almost 10% from its 0.485 to 0.525 under CEAF metrics, which may be important for some applications. The headroom for improvements in recall is similar to MIX1: 11%-15%.

5.2 Tests under NE-CR Condition

Tests under the NE-CR condition show significantly better improvements: ENCORE improves the F-scores on the ACE NWIRE by 31% and 40% with CEAF and 19% and 41% with B^3 over LBJ and BART respectively. While the comparison with LBJ and BART under the NE-CR condition might not be entirely fair because these systems were not optimized for this condition, we show how the "black box" systems can be successfully re-purposed for a different task.

	Precision	Recall	F-Score
ENCORE	**0.534**	**0.559**	**0.542**
Union	0.251	**0.572**	0.347
LBJ	0.332	0.556	**0.413**
BART	0.340	0.464	**0.388**

Table 5: ACE NWIRE with CEAF under NE-CR

	Precision	Recall	F-Score
ENCORE	**0.495**	**0.483**	**0.476**
Union	0.270	**0.527**	0.349
LBJ	0.369	0.470	**0.401**
BART	0.329	0.369	**0.337**

Table 6: ACE NWIRE with B^3 under NE-CR

6. Discussion and Conclusion

In this paper, we introduced ENCORE - our system for entity co-reference resolution based on the synthesis of the results from off-the-shelf co-reference resolution and named entities extraction products. The preliminary experiments we conducted on two test sets under more standard NP-CR condition show marked improvements in the F-scores ranging from 2% to 27% with significant headroom for further improvement. Under the NE-CR condition, the experiments show even better performance improvements of 20% to 40% in the F-scores over the baselines. Our main contribution is to show how the growing number of "black box" off-the-shelf systems can be leveraged to create fast prototypes with superior performance even for the tasks that deviate from their original purpose. Instead of re-creating the existing methods, we focused our efforts on the analysis of their short-comings. Our first targets were the discrepancies between the underlying primary systems. Our heuristics based on a named entity filter proved to be quite effective. We are currently investigating additional heuristics and machine learning approaches to synthesize the primary systems which would further improve the performance of ENCORE.

7. References

Bagga, A. & Baldwin, B. (1998). Algorithms for Scoring Coreference Chains. In *Proceedings of the International Conference on Language Resources and Evaluation.*

Bengtson, E. & Roth, D. (2008). Understanding the value of features for coreference resolution. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing.*

Culotta, A., Wick, M., Hall, R. & McCallum, A. (2007). First-order probabilistic models for coreference resolution. In *Proceedings of the Annual Conference the North American Chapter of the Association for Computational Linguistics - Human Language Technologies.*

Finkel, J.R. & Manning, C.D. (2008). Enforcing Transitivity in Coreference Resolution. In *Proceedings of the Annual Conference the Association for Computational Linguistics - Human Language Technologies.*

Haghighi, A. & Klein, D. (2009). Simple Coreference Resolution with Rich Syntactic and Semantic Features. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing.*

Haghighi, A. & Klein, D. (2007). Unsupervised Coreference Resolution in a Nonparametric Bayesian Model. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics.*

Luo, X. (2005). On coreference resolution performance metrics. In *Proceedings of the International Conference on Language Resources and Evaluation - Human Language Technologies.*

McCallum, A. & Wellner, B. (2003). Towards conditional models of identity uncertainty with application to proper noun coreference. In *Proceedings of International Joint Conference of Artificial Intelligence Workshop on Info Integration on the Web.*

Mitchell, A., Strassel, S., Przybocki, M., Davis, J., Doddington, G., Grishman, R., Meyers, A., Brunstein, A., Ferro, A., and Sundheim, B. (2003). *ACE-2 Version 1.0.* Linguistic Data Consortium, Philadelphia.

Ng, V. (2005). Machine learning for coreference resolution: From local classification to global ranking. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics.*

Ng, V. (2008). Unsupervised Models for Coreference Resolution. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing.*

Ng, V. & Cardie, C. (2002). Improving machine learning approaches to coreference resolution. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics.*

Shah, R., Lin, B., Gershman, A. & Frederking, R. (2010). SYNERGY: A Named Entity Recognition System for Resource-scarce Languages such as Swahili using Online Machine Translation. In *Proceedings of International Conference on Language Resource and Evaluation Workshop on African Language Technology.*

Soon, W.M., Ng, H.T., & Lim, D.C.Y. (2001). A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics,* Volume 27, Issue 4 (December 2001).

Versley, Y., Ponzetto, S.P., Poesio, M., Eidelman, V., Jern, A., Smith, J., Yang, X., & Moschitti, A. (2008). BART: A Modular Toolkit for Coreference Resolution. In *Proceedings of the Annual Conference the Association for Computational Linguistics - Human Language Technologies.*

Vilain, M., Burger, J., Aberdeen, J., Connolly, D., & Hirschman, L. (1995). A model-theoretic coreference scoring scheme. In *Proceedings of Message Understanding Conference-6.*

Concord - A Tool that Automates the Construction of Record Resolution Systems

Christopher Dozier, Hugo Molina-Salgado, Merine Thomas, Sriharsha Veeramachaneni

Thomson Reuters Research and Development
Eagan, MN 55123, USA
chris.dozier@thomsonreuters.com

Abstract

We describe an application we created called Concord that enables software engineers to build and execute Java based record resolution systems (RRS) quickly. Concord allows developers to interactively configure a RRS by specifying match feature functions, blocking functions, and unsupervised machine learning methods for a specific resolution problem. From the developer's defined configuration parameters, Concord creates a Java based RRS that generates training data, learns a matching model and resolves the records in the input files. As far as we know, Concord is unique among RRS generators in that it allows users to select feature functions which are customized for particular field types and in that it allows users to create matching models in a novel unsupervised way using a technique called surrogate learning.

1. Introduction

Record resolution systems match records in one file to records in another file. They are often used to link records pertaining to people and organizations. Record resolution systems play a central role in resolving data across businesses, government agencies, and other organizations. In addition, record resolution systems play a key role in creating cross document coreference systems when structured records for an entity are extracted from text and matched to entity authority files (Dozier and Haschart, 2000) .

This paper describes an application we created called Concord that enables software engineers to build Java based RRSs quickly. Concord allows users to interactively configure a RRS by specifying match feature functions, blocking functions, and unsupervised machine learning methods for a specific problem. From the developer's defined configuration parameters, Concord creates a Java based RRS that generates training data, learns a matching model, and resolves the records in the input files. As far as we know, Concord is unique among RRS generators because it can be easily reconfigured to tackle a variety of record resolution problems by allowing users to select feature functions that are customized for particular field types and by creating matching models in a novel unsupervised way using a technique called surrogate learning.

Concord also supports interactive analysis of the input files and the output matches for a specified resolution system.

2. Overview of Concord

The steps a developer must follow to create a resolution system with Concord are described below. For purposes of our discussion, we assume that we wish to match the records in a File 1 to the records in a File 2.

First, a developer must analyze and align data in File 1 and File 2 records. This means that the developer must determine which fields in File 1 records are semantically compatible with fields in File 2 records. The input analysis part of Concord helps with this step and is briefly described in section 4.

Second, a developer must define feature functions between fields in File 1 and File 2 records. Feature functions are used to determine whether a File 1 and File 2 record pair match. A feature function typically compares one or more fields in a File 1 record with one or more fields in a File 2 record and computes a similarity score.

Concord provides a library of feature functions that can be selected and attached to field elements in File 1 and File 2 to produce feature function values for inclusion in a feature vector. We will discuss this in section 4.

Third, a developer must specify a blocking function that accepts information from a File 1 record and returns a candidate set of File 2 records where the candidate set very probably contains the matching record from File 2 if such a record exists. Concord provides an interactive way to specify a blocking function. We will discuss this in section 4.

Fourth, a user must specify a surrogate labeling function that can automatically label feature function vectors for use in training the RSS matching function. Surrogate labeling functions are functions that apply a normalized value between 0.0 and 1.0 to a feature vector in such a way that the high and low values of the surrogate correlate well with ground truth match and mismatch judgements respectively. We call the function a 'surrogate' because the output of the function serves as a surrogate for manual judgments. We have found that the reciprocal of the block size associated with a given File 1 and File 2 record pair is often a good surrogate function. We will discuss this in section 4.

Fifth, the user must select the amount of training data the Concord system should generate for the machine learner. And the user must select the type of machine learner to use. Concord allows users to choose from a variety of machine learning techniques including a support vector machine with a linear kernel, an SVM with polynomial kernel, and an SVM with a RBF kernel. We will discuss this in section 4.

Sixth, the developer must instruct the Concord system to use the parameters specified in the previous steps to train a matching model for resolving File 1 and File 2 records and to use the model to perform the actual resolution. The resolution is performed by reading each File 1 record, retrieving a set of File 2 records using the blocking function, scoring each feature function vector associated with each File 1 and File 2 pair from the block, and writing to an output file the highest scoring record pair in a block. We will discuss this in section 4.

Finally, the developer must review the resolutions created between File 1 and File 2 records and select upper and lower threshold scores. Record pairs associated with a score above the upper threshold are considered matches that require no editorial review. Record pairs with a score below the lower threshold are considered to represent File 1 records that cannot be matched to File 2 because the highest scoring File 2 record is a mismatch. Record pairs with scores occurring between the upper and lower threshold are considered to be those that require editorial review for accurate categorization into match or mismatch classes. We will discuss this in section 4.

Figure 1 shows a flow diagram that Concord follows to resolve file records. The Concord user interface for resolution system specification is shown in Figure 2.

In the sections that follow, we describe similarities and differences between Concord and other entity resolution tools. We then describe the user interface that Concord provides for specification, configuration, training, and running a resolution system. And we describe Concord's input file analysis tool and its match output analysis tool.

3. Other Work

The only other system that we know of that creates record resolution software is the Febrl system (Christen, 2008). The name Febrl is an acronym standing for Freely Extensible Biomedical Linkage. The Febrl system is written in python and generates python record resolution code. The Febrl system provides utilities to analyze the data in file record fields, provides utilities to specify blocking functions, provides a library of feature function routines, and provides a machine learning based method of performing record resolution. It also has utilities to clean and normalize data.

The main differences between Febrl and Concord are these. Concord provides an unsupervised method of training a matching function using surrogate labeling, while Febrl bases its unsupervised learning method on a seeded nearest neighbor approach. Concord produces Java code rather than python code. Concord provides a set of feature functions that are customized for particular semantic field types such as person names, street addresses, locations, and company names. Concord feature functions couple field normalization with field comparisons in many cases. And Concord is designed to be able to process much larger files than Febrl is.

4. Creating a Resolution System

Concord provides a quick way to build file record resolution systems. Part of the user interface for this system is shown in Figure 2.

Concord provides facilities to specify the names and locations of the files to resolve. Concord also displays information about the composition of the files and allows for the specification of match feature functions, blocking functions, a surrogate function for automatically labeling training data, and a machine learning technique to use to model matching functions.

Concord also allows developers to invoke the training of matching models and the resolution of the records in the files specified.

4.1. Analysis of Input Files

The Concord interface allows the user to specify the names of the files to resolve (called here File 1 and File 2). An example of this can be seen in Figure 2 under *Input Files*.

Concord requires that the files contain a single record per line and that the fields in the record be separated with a '|' character.

The first record in the file must contain the semantic labels of the record fields. And the first field of each record must contain a unique record id.

Once the user has specified File 1 and File2, the resolution system displays to the user the field labels and field values from the two files. Concord also calculates the density and uniqueness of the record field values.

$$density = \frac{p}{n}$$
$$uniqueness = \frac{u}{p}$$

where $n =$ number of records in file, $p =$ number of populated fields across all records for a particular field, and $u =$ the number of unique field entries across all records for a particular field.

An example of the Concord input analysis is shown in Figure 2 in *Field Description* section of the screen. We can see from Figure 2 that File 1 and File 2 have 862,962 and 837,750 records respectively. We also see that the first field label in each file is *PERSON-ID* and that these fields have density and uniqueness values of 1.0. This indicates that the *PERSON-ID* fields are fully populated and unique throughout the files.

4.2. Selection of Feature Functions

Resolution systems typically use a set of feature functions to determine how well a File 1 and File 2 record pair match. Feature functions work by comparing a field from the File 1 record with a field from the File 2 record and returning a similarity score for the field pair. More formally expressed, a feature function is the following:

$$F_n(field_{record_i}, field_{record_j}) = x_n$$

where F_n compares $field_{record1}$ (a field from a File 1 record) with $field_{record2}$ (a field from a File 2 record) to compute a value x_n. x_n is an element of the feature vector

$$\vec{x_{i,j}} = \{x_1, x_2, x_3 ... x_m\}$$

that is used to compute a comparison score between a File 1 record *i* and a File 2 record *j*. That is,

$$MatchScore(record_i, record_j) =$$
$$ScoringFunction(\vec{x_{i,j}})$$

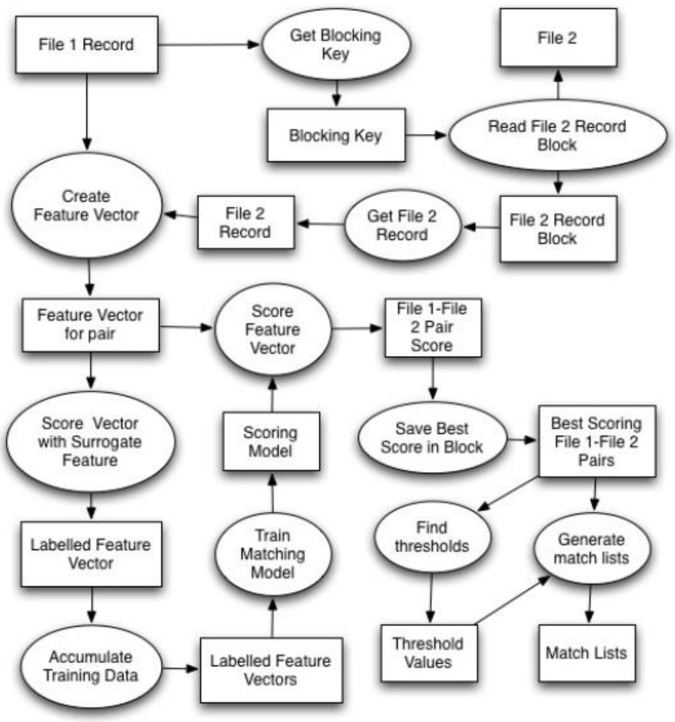
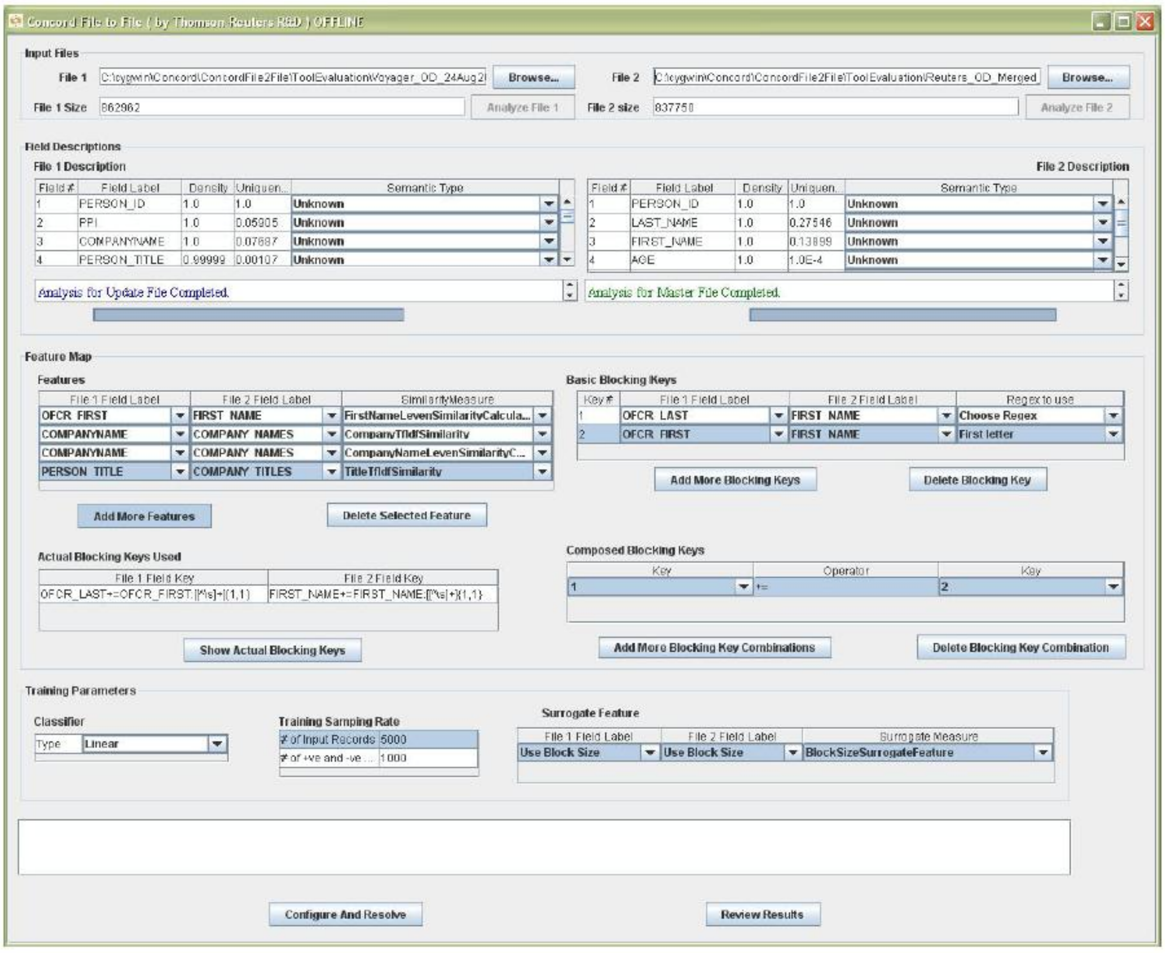


Figure 1: Concord Processing Flow



Input Files

File 1: C:\cygwin\Concord\ConcordFile2File\ToolEvaluation\Voyager_OD_24Aug2

File 1 Size: 862962

File 2: C:\cygwin\Concord\ConcordFile2File\ToolEvaluation\Reuters_OD_Merged

File 2 size: 837750

Field Descriptions

File 1 Description

Field #	Field Label	Density	Uniquen...	Semantic Type
1	PERSON_ID	1.0	1.0	Unknown
2	PPI	1.0	0.05905	Unknown
3	COMPANYNAME	1.0	0.07687	Unknown
4	PERSON_TITLE	0.99999	0.00107	Unknown

Analysis for Update File Completed.

File 2 Description

Field #	Field Label	Density	Uniquen.	Semantic Type
1	PERSON_ID	1.0	1.0	Unknown
2	LAST_NAME	1.0	0.27546	Unknown
3	FIRST_NAME	1.0	0.13899	Unknown
4	AGE	1.0	1.0E-4	Unknown

Analysis for Master File Completed.

Feature Map

Features

File 1 Field Label	File 2 Field Label	SimilarityMeasure
OFCR_FIRST	FIRST_NAME	FirstNameLevenSimilarityCalcula...
COMPANYNAME	COMPANY_NAMES	CompanyTfIdfSimilarity
COMPANYNAME	COMPANY_NAMES	CompanyNameLevenSimilarityC...
PERSON_TITLE	COMPANY_TITLES	TitleTfIdfSimilarity

Basic Blocking Keys

Key #	File 1 Field Label	File 2 Field Label	Regex to use
1	OFCR_LAST	FIRST_NAME	Choose Regex
2	OFCR_FIRST	FIRST_NAME	First letter

Actual Blocking Keys Used

File 1 Field Key	File 2 Field Key
OFCR_LAST+=OFCR_FIRST:[[^\s]+]{1,1}	FIRST_NAME+=FIRST_NAME:[[^\s]+]{1,1}

Composed Blocking Keys

Key	Operator	Key
1	+=	2

Training Parameters

Classifier

Type	Linear

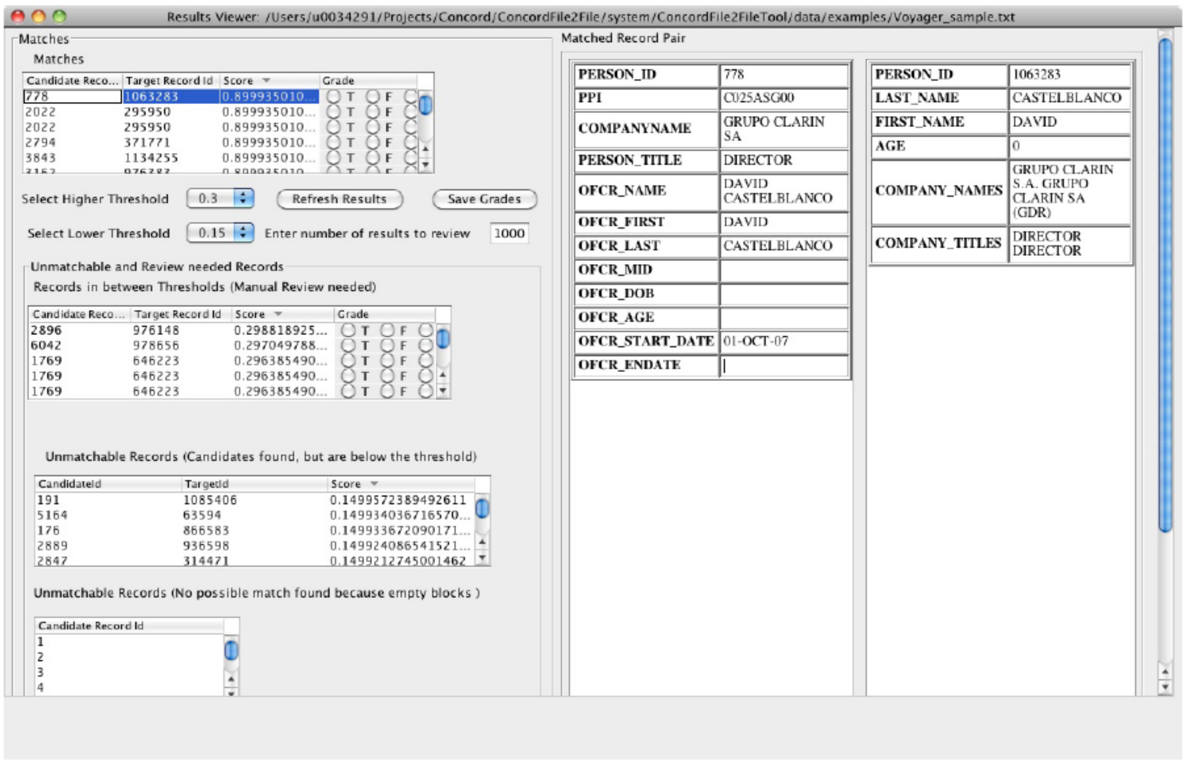
Training Samping Rate

# of Input Records	5000
# of +ve and -ve ...	1000

Surrogate Feature

File 1 Field Label	File 2 Field Label	Surrogate Measure
Use Block Size	Use Block Size	BlockSizeSurrogateFeature

Figure 2: Concord Interface


Results Viewer: /Users/u0034291/Projects/Concord/ConcordFile2File/system/ConcordFile2FileTool/data/examples/Voyager_sample.txt

Matches

Matches

Candidate Reco...	Target Record Id	Score	Grade
778	1063283	0.899935010...	◯ T ◯ F ◯
2022	295950	0.899935010...	◯ T ◯ F ◯
2022	295950	0.899935010...	◯ T ◯ F ◯
2794	371771	0.899935010...	◯ T ◯ F ◯
3843	1134255	0.899935010...	◯ T ◯ F ◯

Select Higher Threshold 0.3

Select Lower Threshold 0.15

Enter number of results to review 1000

Unmatchable and Review needed Records

Records in between Thresholds (Manual Review needed)

Candidate Reco...	Target Record Id	Score	Grade
2896	976148	0.298818925...	◯ T ◯ F ◯
6042	978656	0.297049788...	◯ T ◯ F ◯
1769	646223	0.296385490...	◯ T ◯ F ◯
1769	646223	0.296385490...	◯ T ◯ F ◯
1769	646223	0.296385490...	◯ T ◯ F ◯

Unmatchable Records (Candidates found, but are below the threshold)

CandidateId	TargetId	Score
191	1085406	0.1499572389492611
5164	63594	0.149934036716570...
176	866583	0.149933672090171...
2889	936598	0.149924086541521...
2847	314471	0.1499212745001462

Unmatchable Records (No possible match found because empty blocks)

Candidate Record Id
1
2
3
4

Matched Record Pair

PERSON_ID	778
PPI	C025ASG00
COMPANYNAME	GRUPO CLARIN SA
PERSON_TITLE	DIRECTOR
OFCR_NAME	DAVID CASTELBLANCO
OFCR_FIRST	DAVID
OFCR_LAST	CASTELBLANCO
OFCR_MID	
OFCR_DOB	
OFCR_AGE	
OFCR_START_DATE	01-OCT-07
OFCR_ENDATE	

PERSON_ID	1063283
LAST_NAME	CASTELBLANCO
FIRST_NAME	DAVID
AGE	0
COMPANY_NAMES	GRUPO CLARIN S.A. GRUPO CLARIN SA (GDR)
COMPANY_TITLES	DIRECTOR DIRECTOR

Figure 3: Concord Output Analyzer

Concord allows for the easy selection of feature functions. The user selects a feature function by specifying the File 1 and File 2 fields to compare and then selecting a comparison function from a drop down menu to apply to the fields. Table 1 shows how three feature functions would be specified for matching corporate officer records from two different files. The section of the Concord interface in Figure 2 labelled *Feature Map* is where the developer can specify feature functions.

Concord provides the user with a large set of feature functions as well as the ability to add specialized functions the user may require for particular resolution problems.

The feature functions Concord provides are of five types.

The first type are string comparison functions that are customized to particular semantic types. These functions include compareFirstNamesLevenschtein(), comparePhoneNumbers(), compareCityState(), compareZipCode(), and compareStreetAddress(). These comparison functions combine string edit distance rules with knowledge of naming conventions associated with particular name types. For example, the compareFirstNamesLevenschtein() feature function gives high match scores to names that have compatible nicknames. Without such a nickname check, many compatible nickname pairs (e.g., *William* and *Bill*) would be separated by large edit distance and would thus have low feature match scores.

The second type are tf-idf cosine comparison functions whose idf term values are local word level frequencies associated with particular fields in File 1 or File 2. An example of this is a companyLocalTF-IDF cosine similarity function that uses the frequency counts associated with company name fields in the File 1 and File 2 files specified.

The third type of feature function are tf-idf cosine comparison functions whose idf term values are taken from word level frequencies associated with large entity name lists outside the framework of a particular resolution problem. An example of this is a companyGlobalTF-IDF cosine similarity function that uses the frequency counts associated with large company name lists that have been assembled from company authority files. So, in this case, we might use a Reuters Company database to compute idf values to use to compare company names.

The fourth type of function are those in the family of string similarity functions. They includes among others the Hamming distance, Levenshtein distance, Smith-Waterman distance, and Jaro-Winkler distance. Each of these functions measures the distance of one string to another as function of the number of character changes that need to be made to transform one string into another. These functions are often robust across different semantic types.

The fifth type of function are specialized idiosyncratic functions a user needs for particular resolution problems. In these cases, Concord allows users to write their own comparison feature functions and add them to an extensible library of special functions. An example of this type of function might be a function that compares proprietary product codes across two files within a particular company.

By providing users with standard comparison functions for strings and common semantic field types as well as providing the capability of adding specialized functions, Concord strikes a balance between enabling the reuse of standard functions and the customization of functions that may be required for particular resolution systems.

File 1 Field	File 2 Field	Comparison Function
OFCR-FIRST-NAME	FIRST-NAME	FirstNameLevenschtein
COMPANY-NAME	COMPANY-NAMES	CompanyTfIdf-Distance
COMPANY-NAME	COMPANY-NAMES	Levenschtein

Table 1: Specifications for Three Feature Functions.

4.3. Selection of Blocking Functions

Concord allows for the selection of blocking functions. The purpose of blocking is to increase the efficiency of resolution by limiting the number of File 2 records to which we must compare a File 1 record in order to find a match. A good blocking function will return a small mean number of File 2 records for each File 1 record under consideration and will also return within the block with a high probability the File 2 record that best matches any given File 1 record. A blocking function works by constructing a File 1 block key from fields in a File 1 record and reading all File 2 records that can be indexed by that blocking key value. The blocking key indexes for File 2 are build by constructing keys from each File 2 record using fields in the File 2 records that are compatible with the fields used to create File 1 block keys.

An example of a blocking key could be the contents of the last name field in a File 1 record file of corporate officers and the last name field in a File 2 record file of corporate officers.

Concord allows users to specify blocking keys in two steps. First, the user specifies a set of basic blocking keys. A basic blocking key shows how a single field or part of a field from File 1 should be matched against a single field or part of a field from File 2. The second part of the blocking specification shows how the basic blocking keys should be assembled to form a final blocking key.

An example of basic and final blocking keys is shown in the interface illustration in figure 2. Here the basic blocking keys are last name and first initial of the first name. The first basic block key is last name and consists of the field OFCR-LAST from File 1 and the field LAST-NAME in File 2. The second basic block key is first initial of first name and consists of first character of field OFCR-FIRST in File 1 and first character of field FIRST-NAME in File 2. These two basic block keys are then concatenated to form a final block key; namely, last name plus first letter of first name. The composition of the final block key in our example is shown in figure 2 in the screen panel labeled *Actual Blocking Keys Used*.

4.4. Selection of Surrogate Function for Training

The system allows for the selection of a surrogate training function. Surrogate training functions are used to label training feature vectors in lieu of manual judgments (Veeramachaneni and Kondadadi, 2009). Developers can select surrogate functions from the drop down menu labelled *Surrogate Feature* under the section of the user interface labeled *Training Parameters*.

Surrogate functions must have the characteristic that high values returned by the function correlate on average with true positives (matches) while low values correlate on average with true negatives (mismatch). Also the surrogate function must be class-conditionally independent of the other feature functions.

In our experience, a particularly useful surrogate feature is the reciprocal of the block size associated with a particular feature vector set, i.e.,

$$surrogateFeature = \frac{1}{block-size}$$

This works well because in small blocks the ratio of the positive pair to negative pairs is large, the surrogate score is large, one true positive is generated, and a small number of negatives are generated. While for large blocks, the ratio of the positive pair to the negative pairs is small, the surrogate score is small, one true positive is generated, and many negatives are generated. So the mean surrogate score for positive pairs is larger than the mean score for negative pairs. For instance, for a singleton block, the surrogate score would be 1.0 for the single positive feature vector. While, for a block containing 100 records, 99 true negative feature vectors would have a score of 0.01 and 1 true positive feature vector would have a score of 0.01.

The graph in Figure 4 shows the correlation between the reciprocal block size surrogate feature and the company name tf-idf feature for a randomly selected set of record pairs from two different corporate officer files blocked on last name of officer. True positives (matching pairs) are indicated with an $\circ$ and true negatives (mismatching pairs) are indicated with a $\triangle$. We can see from the distribution of $\circ$ and triangles that true negatives are associated with both low scoring surrogate features and low scoring company name tf-idf feature values while true positives are much more likely to be associated with high surrogate and company tf-idf scores.

It turns out that $P(y = 1|x_2)$ is a monotonically increasing function of $E[x_1|x_2]$ for surrogate scores based on the inverse of the block size in many circumstances. Here x_1 is the surrogate inverse block size score and x_2 is the feature vector associated with a record pair. So we can build a regression SVM for $E[x_1|x_2]$ itself which will rank results in the same order as a classifier that modelled $P(y = 1|x_2)$ directly. We describe this more fully in (Veeramachaneni, 2009).

4.5. Training System

Once feature functions, the blocking function, and the surrogate function have been specified, the developer must specify a machine learner and sampling parameters for the generated labeled training feature vectors.

For machine learners, Concord offers developers the choice of regression SVMs with linear, polynomial, and RBF kernels. The developer can choose the machine learner from the drop down menu labelled *Classifiers* under the section of the user interface labeled *Training Parameters*.

Concord samples training feature vectors by randomly choosing a set of *n* File 1 records, generating a pool of labeled feature vectors, and then randomly selecting *m* labeled feature vectors from the generated pool.

The number *n* and *m* are selected by the user under the *Training Samples Rate* screen label in Figure 2.

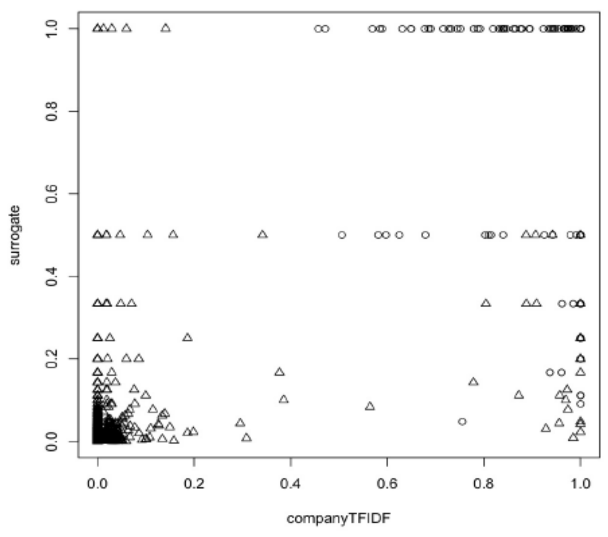


Figure 4: Graph of Training Data Showing Correlation Between the Reciprocal Block Size Surrogate Feature and the Company Name TF-IDF Feature. ○ is match. △ is mismatch.

File 1 ID	File 2 ID	Highest Match Score in Block
123	55301	0.4501
439	67012	0.1534
871	12082	0.8921

Table 2: Format and Example of Resolution Records

4.6. Record Resolution

The user activates training and resolution of the records in Concord by clicking on a screen button labeled *Train and Resolve*. This causes Concord system to create a matching model using the training feature vectors and then to resolve each of the File 1 records to File 2.

Concord performs the resolution of each File 1 record by reading a block of File 2 records using the blocking key and scoring each File 1 and File 2 record pair in the block using the matching model. The record pair associated with the highest scoring feature vector is stored in the Concord output resolution file. The format of three resolution records is shown in table 2.

A Concord user can browse the resolution results via an output analyzer screen. Figure 3 shows an output analyzer display for the results obtained when we resolved two files containing records describing corporate officers and directors. In this case, File 1 contains 862,962 records and a File2 contains 837,750 records. The resolution parameters governing this resolution system are shown in Table 3.

Figure 3 shows four ranked result lists. The first list shows File 1 and File 2 match pairs ordered by match scores that are above 0.3. The second list shows match pairs with scores between 0.3 and 0.15. The third list shows match pairs with scores below 0.15. The fourth list shows the ids of File 1 records for which no File 2 candidate records were found with the blocking key.

Selection Type	Selected Parameter	Description
File 1	Voyager File	862,962 records
File 2	Reuters File	837,750 records
Feature Function 1	CompareFirstName	edit distance
Feature Function 2	CompareCompanyName	tf idf
Feature Function 3	CompareCompany Name	edit distance
Feature Function 4	CompareAge	
Feature Function 5	CompareTitle	edit distance
Blocking Key	LastName + FirstInitialFirstName	
Surrogate Function	BlockSize	
Machine Learner	SVM Linear Regression	

Table 3: Configuration Parameters for RSS for Officers and Directors Resolution Experiment

Match Score Threshold	Precision	Recall
0.10	0.833	0.966
0.20	0.972	0.852
0.30	0.995	0.808
0.40	0.995	0.800
0.50	1.00	0.792
0.60	1.00	0.761
0.70	1.00	0.709
0.80	1.00	0.665

Table 4: Resolution Precision and Recall for Corporate Officers and Directors Problem

The user may scroll through each of these lists and see details about the matched records by clicking on the row list. In the figure, the user has clicked on the top scoring record pair in the top list. The screens to the left show detail about the matched pair File 1 record 778 and File 2 record 1063283. Both records pertain to a director at *Grupo Clarin SA* named *David Castelblanco.*

The upper and lower display thresholds can be set by the user.

5. Evaluation

We evaluated our system on a number of real world record linkage problems. An example of one of these systems is the merging of corporate officers and directors data we discussed in previous sections.

In our example system, we matched 862,961 File 1 records to 837,760 File 2 records. The configuration parameters for the system are shown in Table 3.

To assess precision and recall of the system, we checked 300 randomly selected highest scoring match pairs per block.

Figure 5 shows how match scores vary by highest scoring record pair. A '+' symbol indicates that pair is truly a match. A △ symbol indicates that pair is a mismatch. Table 4 shows how the precision and recall of the system vary as the match score belief threshold varies. Here any record pair whose score falls above the threshold is deemed a match and any File 1 record whose best match to File 2 falls below the threshold is deemed unmatchable to a File 2 record.

In our example, using a threshold of 0.3, we created a system that resolved records with a precision of 0.99 and a recall of 0.80.

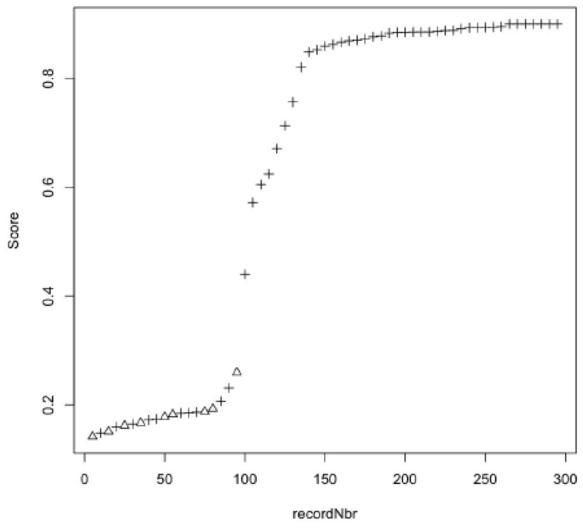


Figure 5: High Match Scores for 300 Officers Records. + is true match. △ is mismatch.

6. Discussion

Concord works well for file resolution problems for two main reasons. First, many file matching tasks can be solved by applying common field value comparison routines (feature functions). Second, in many cases, the files can be resolved automatically using a technique we call 'surrogate learning' (Veeramachaneni, 2009).

Surrogate learning is an unsupervised machine learning technique that uses a surrogate feature function in place of manually produced classification judgments to create training data. Surrogate feature functions must be class conditionally independent of other feature functions and should have the property that their mean value for matches is greater than their mean value for non-matches.

A surrogate feature that is particularly elegant for entity resolution problems is the reciprocal of the block size. Using this feature as a surrogate will work if the block size variance is relatively large and if the probability of the block containing the ground truth matching resolution record is high.

Future work on Concord will include development of utilities to assist the developer in finding optimal feature function sets and utilities to assist user in selecting optimal match threshold values. Future work will also include the search for robust feature functions usable in a wide range of entity resolution tasks.

7. Conclusion

This document describes an application called Concord that enables software engineers to build and execute Java based record resolution systems quickly. Concord allows developers to interactively configure a record resolution system by specifying match feature functions, blocking functions, and unsupervised machine learning methods for a specific problem. From the developer's defined configuration parameters Concord creates a Java based record resolution system, that generates training data, learns a matching model and resolves the records in the input files. As far as we know, Concord is unique among record resolution system generators because it can be easily reconfigured according to the problem by allowing users to select feature functions that are customized for particular field types and by creating matching models in an unsupervised way using a technique called surrogate learning.

8. References

P. Christen. 2008. Febrl - a freely available record linkage system with a graphical user interface. *In Proceedings of Second Australasian Workshop on Health Data and Knowledge Management Current Anthropology. Wollongong, NSW, Australia.*

C. Dozier and R. Haschart. 2000. Automatic extraction and linking of person names in legal text. *In Proceedings of RIAO 2000 (Recherche d'Information Assistee par Ordinateur). Paris, France*, pages 1305–1321.

S. Veeramachaneni and R. Kondadadi. 2009. Surrogate learning from feature independence to semi-supervised classification. *In NAACL Workshop on Semi-Supervised Learning. Boulder, Colorado, USA*.

S. Veeramachaneni. 2009. Surrogate learning for record linkage using inverse block size. *In Thomson Technical Report 2009-1345, Eagan, Minnesota, USA*.

Identification, Extraction and Population of Collective Named Entities From Business News

Brett Drury and J. J. Almeida

LIAAD-INESC and University of Minho

Abstract

Sentiment analysis of business news has become an increasingly popular area of research for both the practitioner and academic. The future financial prospects of companies can be estimated through the aggregation of sentiment over a period of time. The aggregation of sentiment for a specific company is only possible if the company is explicitly mentioned in the news text. In certain instances, news text may refer to groups or collections of companies, for example ”The Automotive Sector” or ”The Russell Group of Universities”. Widely available named entity dictionaries will not recognize these groups of companies, and consequently, it may not be possible to assign sentiment attributed to these groups of companies to their individual members. This paper describes a method for identifying groups of companies, which for the purposes of this paper will be known as ”Collective Entities”. The described method is corpus based: it uses linguistic patterns to identify Collective Entity Names, their members and their natural relations with other Collective Entities. The described methodology contains the following steps: 1. Identify and validate seed extraction patterns, 2. Expand seed patterns, 3. Extract and validate Collective Named Entities, 4. Extract related Collective Named Entities, 5. Construct and populate an Ontology and 6. Expand the members of Collective Entity sets with Linked Data.

1. Introduction

Sentiment analysis of business news stories has become an increasingly popular area of research. The ”emotion” of a phrase or sentence can be assigned to an entity in the story. The phrase, ”Microsoft faces a bleak 2010”, contains the adjective ”bleak”, which is negative. The predefined value for the word ”bleak” can be assigned to the entity ”Microsoft”. An aggregation of sentiment may provide an indication of its future financial prospects. It is common for journalists to refer to groups of companies (”Collective Entities”) in their news stories. The hypothetical phrase, ”The car industry faces a bleak 2010”, contains a Collective Entity, ”The car industry”. At the present time widely available named entity dictionaries will fail to identify this named entity and consequently any sentiment information cannot be assigned to the Collective Entity or its members. There is a further problem: there is no commonly accepted comprehensive list of Collective Entities[1] which a journalist can refer to, consequently journalists may a apply a number of different labels to the same Collective Entity. For example, a journalist may refer to companies who produce cars as: ”The Car Industry”, ”The Automotive Business”, ”The Car Construction Business” or ”The Automotive Fabrication Sector”. The failure to identify Collective Entities may result in the incorrect assignment of sentiment which may result in flawed financial inferences.

This paper describes an attempt to construct a ”Collective Entity” (CE) resource for the GATE Framework. (Cunningham, 2002) The resource is limited to companies and their containing CE. There may be other ”Collective Entities”(CEs) in business news which have members which are not companies; however, the identification of these entities was beyond the scope of this paper.

The derived resource will be used to assign sentiment found in news stories to CEs and their members. The production of these resources relied upon a corpus based method which learnt CE titles from a large document collection. The CE titles were initially described as a series of lists. An ontology of CEs and its members was constructed to provide a richer description.

2. Identification of Collective Entities

A ”Named Entity” is a term for which there are one or many rigid designators (Kripke, 1982). A rigid designator can be proper names as well as certain natural kind terms like biological species and substances (Nadeau, 2007). In this paper the definition of a rigid designator was expanded to include inconsistently applied designators which are in common use in formal written English. There may be many ”inconsistent designators;” however, for the purposes of this paper, a legitimate inconsistent designator is a name or term which is frequently used in the news media to describe a collection of more than one company. It is possible that a number of these CEs may have very descriptive and obvious names, but there are a larger number of CEs whose titles contain vague language or jargon, for example ”The CRM Business” or ”The DME Industry”. ”Arbitrary names” which have been formally registered have been excluded from this study, for example company names and governmental or non-governmental agencies .

2.1. Named Entity Extraction Methodology

The described approach used linguistic patterns to identify CEs. This technique was preferred to the more popular approaches of identifying named entities with supervised machine learning techniques (Nadeau, 2007) because supervised machine learning may require large amounts of

[1] A number of financial newspapers group shares of companies by business area, however these groupings are not generally referred to by journalists. In addition the groupings will only refer to companies listed on that exchange

labelled data which can be a labour intensive task. In addition, labelled data may contain an unrealistic representation of the target entities (McEnery, 1996) and consequently the supervised technique may return a sub-optimal number of CEs. It was expected that the simplicity of the approach would be counterbalanced by the "unreasonable effectiveness of data" (Halevy, 2009) of a large document collection.

2.2. Data Preparation

The experiments were undertaken with the GATE framework (Cunningham, 2002) . GATE is a collection of Java Classes produced by the University of Sheffield which can be used to process text. GATE contains a large number of resources; however, the described experiments used two resources, the ANNIE Gazetteer and JAPE. JAPE allows the manipulation of annotations in text which have been produced by other GATE resources.

Corpus Construction

A large collection of news stories was required for the GATE experiments. The news stories were retrieved from freely available sources on the internet. The proposed experiments required that the headline, description and story text were recorded. The stories were sent to the OpenCalais web service (Reuters, 2010) for the addition of meta data.[2]

Expansion of the GATE Gazetteer

The ANNIE Gazetteer contained a number of finite lists for named entities. The company named entity list contained a small number of company names. This was an inadequate representation of the companies in the document collection. An aim of this paper was to construct a formal representation of CEs and their members which are companies; consequently, it was necessary to greatly expand the company list. The OpenCalais (Reuters, 2010) web service uses a series of rules to identify named entities, and consequently, it extracted a comprehensive list of companies from the document collection meta-data. This rule based approach often produced errors. The erroneous company names appeared infrequently, and consequently, it was required that each candidate company name have a minimum frequency in the document meta-data. The higher the frequency of the company name, the higher the probability that the name was correct. A total of 42,823 company names were extracted.

2.3. Identifying Seed Linguistic Patterns for Extraction of Collective Entities

The initial experiments required a series of seed patterns to extract an initial sample of CE titles. The first seed patterns were constructed from the intuition of a native English speaker who suggested that the word "industry" would be a good indicator of a CE title. This intuition required validation; consequently, a list of common words which were part of known CE titles was created. This list included the words "car", "banking", "finance", etc. This list will be referred to as "The Known CE Variable List". A regular expression was created where one word either side of the words from the Known CE Variable List was extracted. The most frequent words identified by the regular expression were not part of a CE title. The only exception was the word industry. This experiment confirmed the intuition of the native speaker. A sample of the words returned by the regular expression can be found in table 1

An expanded list of words equivalent to industry was created. This list will be referred to as "The Industry Synonym List". The similar words were taken from The Oxford Thesaurus Of English (Oxford, 2010). A new regular expression was created to extract three-word phrases where list items from The Industry Synonym List constituted the third word. A manual inspection of the results indicated that a large number of phrases were not CE titles. These phrases contained one or more of the following: stop words, numbers, continuations, adjectives, verbs, comparisons, and quantifications. Therefore, a number of new regular expressions were created, which were variations of the previously described pattern. The variant patterns contained one or more refinements. Table 2 describes the regular expression refinements as well as its subsequent precision and recall.

2.4. Validation of Initial Patterns

The accuracy of each run was measured with the following criteria: precision and pattern recall. It was not possible to verify all of the extracted CE titles because of the large number of candidate phrases; consequently, the precision of each term was calculated by the verification of the 100 most frequent phrases. The candidate CE titles' frequency in the corpus followed a Zip's distribution (LI, 1992) and therefore a low precision amongst the most frequent candidate titles would impair greatly the overall accuracy of the developed resource. It was not possible to calculate a recall figure because there was no exhaustive list to compare the titles against; consequently, a raw pattern recall figure was calculated.

Frequency	Pattern
1046	the ... industry
952	GAAP ... measures
906	the ... sector
815	the ... crisis

Table 1: Extracted Phrase Delimiters for Known Collective Entities

Regular Expression	Precision	No. Recalled
All Words + "the"	0.33	23344
All Words (-"Industry") + "the"	0.36	5033
Industry - "the"	0.72	3753
Industry + "the"	0.98	1450

Table 2: Precision of Collective Entity Titles

All Words = All words which were equivalent to the word "industry"
"the" = First word of the phrase must be "the"

[2] The OpenCalais web service is a product from Reuters which provides semantic meta-data for news stories, for example named-entity identification, news story categorizatio, etc.

2.5. Calculation of Final Collective Entities List

A new regular expression was created for each item in The Industry Synonym List. The regular expression used the most effective refinement described in table 2, which was the word "the" as the first word and the list item as the third word in the three word pattern. The 100 most frequent candidate CE titles were extracted. The second word of each correct candidate (CE variable) phrase was compared to the CE variable of candidate phrases generate by the industry pattern. For example, candidate phrase "The Finance Sector", the word "Finance" would be the CE variable. Each correct CE variable identified by patterns without the word "industry" was also contained in the list of CE variables generated by the industry pattern. The final list was calculated by extracting the correct terms from the industry list and duplicating the phrase for each term. The duplication consisted of replacing the term "industry" with the new term. An example of an expansion is detailed below.
Expansion of Collective Entity Names

$$Car\ Industry = car\ business,\ car\ sector..$$

The above experiments were with single term patterns. A further number of experiments were made with multi-word patterns. The patterns identified the word "the" and up to three words and then a word from The Industry Synonym List ("industry", "business" etc). There were substantially less extracted terms, and these candidate terms were hand checked. The terms were expanded in the same manner as the single term candidate phrases.

2.6. Identification of Related Collective Entities

A number of the previously extracted CEs had natural relations, for example "The Oil and Gas Industry". Relations between CEs may be important as it may indicate a subtle method of assigning sentiment. For example, The Oil and Gas Industries are related, and consequently, sentiment in a news story about one industry may, in certain circumstances, be attributed to the other.

The related CEs were identified with the following pattern: Valid CE, the word "and", Valid CE. This pattern returned 9380 phrases. A number of relations were repeated, for example "The Oil and Gas Industry" and "The Gas and Oil Industry".

3. Ontology Construction and Population

The motivation of this work was not only to identify CE titles, but also to identify their members and represent their natural relations with other CEs. An ontology can represent CEs and their relations and members. It can also form the basis of descriptive gazetteers which provide detailed annotations of named entities in text.

3.1. Identification of Collective Entities Members

The members of a CE were companies. There were no comprehensive lists of CE members; consequently, it was necessary to learn the members of each CE. The corpus was processed with the new gazetteer CEs and related CEs lists as well as the expanded company list. The location of each CE and company was recorded.

A number of population experiments were conducted. The experiments were based upon the following criteria: 1. Company Proximity to the CE, 2. Companies Contained in a News Story with a CE in the Headline Text. The proximity experiment identified companies which were in the same sentence as a CE. There were a number of identifiable situations where companies in close proximity to the CE needed to be excluded because they were not members. This was because they were frequently a 3rd party commenting on the CE; for example, the phrase "An analyst from Panmure Gordon" (Pignal, 2008) would indicate that the company Panmure Gordon should be excluded. It was possible to annotate the text to exclude companies which had a high probability of not belonging to the CE. This annotation was achieved with two JAPE' rules, the rules were:

- Rule 1: Pattern = A term which indicate a 3rd party (Consultant, Analyst etc), a word of belonging ("of","from", etc.) and a company name.
- Rule 2: Patterns = Possessive form of a company and a person's name.

The second experiment utilized the observed phenomenon that headlines provide a very good indication of the content of the news story (Andrew, 2007). News stories with headlines which contained a CE were extracted and all the companies in the text were assigned to the headline CE. The validation of the experiments were with a precision measure as well as the number of populated CEs. The precision measure was calculated by extracting 100 entity classes and verifying the legitimacy of their members.

Experiment Type	Precision	Raw Collective Entity Recall
Headlines	0.66	630
SameLine	0.77	3535

Table 3: Collective Entities Population

3.2. Ontology Population

The ontology was constructed using the OWLAPI (OWLAPI, 2010). OWLAPI is a collection of Java classes which allow the construction and manipulation of ontologies. The constructed ontology contained two classes: CE and Company. The names of the individual instances of the CE class were derived by extracting the CE variable(s) which required removing the word,"the" and the CE indicator. For example, The Finance Industry and The Finance Business were represented as an individual with the name "Finance".

The companies may be known by a variety of names. The OpenCalais meta-data provided a disambiguation facility for company names; when it was available, it was used to name the company "individuals". If it was not available, then the company name extracted from the news text was used. The relations between individuals and their data properties is described in table 4. The 64,000 entity limit of the OWLAPI imposed a number of restrictions on the relations between entities; consequently, the relations were "one way".

Property Type	Name of Relation	Relation Between
ObjectPropery	memberof	Companies, CE
ObjectPropery	relatedto	CE, CE
DataProperty	OpenCalaisURI	N/A

Table 4: Ontology Relations

The DataProperty described in table 4 may point to a Linked Data[3] page. A Linked Data page contains more information about the company and in some circumstances the Linked Data page has detailed information about the company's competitors. It was not possible to use this information in the ontology because it would have exceeded the 64,000 entity limit of the OWLAPI. The ontology does provide the URI to the Linked Data page and consequently this information can be accessed by any application which has access to the ontology.

4. Conclusion and Further Work

This work addresses an obvious gap for commonly available named entity dictionaries. The extended GATE Gazetteer lists provide a valuable tool for researchers investigating financial news. The ontology provides a detailed description of the interrelation of some of the CEs as well as detailed representation of their members. A larger number of CE members can be accessed through linked data.

This work will assist researchers and practitioners in the following areas:

- Sentiment and event information attribution to economic actors
- Identification of non-obvious targets for sentiment and event information attribution
- Construction of descriptive ontology based gazetteers
- News story recommendation and information extraction

The resources have a number of limitations. The ontology has errors such as erroneously assigned CE members. The ontology is coarse grained. There are two levels of representation: CE and companies. The CEs can be of various sizes, and consequently, one CE may be a member of another CE.

[3]A more detailed description of Linked Data can be found here: http://www.opencalais.com/documentation/calais-linked-data

The described approach is simple; however, it produced resources which contained a significant number of CEs and their members. It may be possible to use linguistic patterns to identify labelled data for a machine learning technique which may identify additional CEs and their members. It is possible that the derived resource can form the basis of a more sophisticated and accurate representation of CEs and their members.

5. References

H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. Gate: A framework and graphical development environment for robust nlp tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, 2002

Blake C. Andrew. Media-generated shortcuts: Do newspaper headlines present another roadblock for low-information rationality? *The Harvard International Journal of Press/Politics*, 12(2):24–43, 2007.

Alon Halevy, Peter Norvig, and Fernando Pereira. The unreasonable effectiveness of data. *Intelligent Systems, IEEE*, 24(2):8–12, 2009.

Saul Kripke. *Naming and Necessity*. Boston: Harvard University Press., 1982.

W. Li. Random texts exhibit zipf's-law-like word frequency distribution. In *Information Theory, IEEE Transactions on"*, 1992.

Tony McEnery and Andrew Wilson. *Corpus Linguistics*. Edinburgh University Press, 1996.

David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Journal of Linguisticae Investigationes*, 30(1):1–20, 2007.

Oxford Thesaurus of English. Oxford thesaurus of english. http://www.askoxford.com/worldofwords/thesauri/?view=uk, consulted in 2009.

OWLAPI. The owl api. http://owlapi.sourceforge.net/, consulted in 2010.

Stanley Pignal. Stagecoach gains from public transport users. 2008. www.ft.com/cms/s/0/8e3ad9fc-a590-11dd-9d26-000077b07658.html, consulted in 2010.

Reuters. Calais web service. http://opencalais.com/, consulted in 2009.

Name Matching Evaluation Framework

Dmitry Zelenko
SRA International
4300 Fair Lakes Ct.
Fairfax, VA 22033 USA
dmitry_zelenko@sra.com

Abstract

We present an evaluation methodology created as part of designing, developing, and tuning a name matching system. The methodology is particularly suited to improving performance of a single system being developed, in addition to comparison of multiple systems. We address a number of issues encountered in the process such as the need for quick automatic regression testing, incomplete gold standard data, and scalability. An implementation of the proposed methodology is released and available upon request.

1. Introduction

When building a name matching system, one goes through the typical iterative process of making engineering decisions and verifying them experimentally. Verification should highlight whether a decision actually leads to a more accurate system. Unfortunately, accuracy of a name matching system is notoriously difficult to judge, and typical evaluations involve a lot of human effort (Miller *et al.* 2008). Such human involvement is very expensive and not appropriate for rapid system development, where a system designer needs immediate feedback. Therefore, there is a pressing need for an automatic system that obviates human involvement in name matching evaluation and provides for robust quantitative measures of system accuracy.

Accuracy of a name matching system should reflect a typical system application scenario. For example, a user may enter a name as a search query with respect to a database, and would like to see all records with likely variants of the name returned. Another example is the process of database consolidation, where names contained in both databases are required to be matched, and corresponding records are to be merged based on name matching decisions. In general, both the search scenario and the consolidation process usually involve other non-name attributes and are part of the larger process of identity resolution. Our goal here, however, is to evaluate a name matching system *per se*; therefore, we will focus on a typical search scenario involving only names, with no extra attributes.

A search paradigm for name matching evaluation suggests leveraging the evaluation methodology of information retrieval. Indeed, the TREC community developed appealing techniques for comparison of information retrieval systems (Voorhees & Harman 2000). The methodology has been employed, to a large extent, for name matching system comparison as well (Miller *et al.* 2008). We aim to complement the evaluation methodology for name matching system *comparison* with an appropriate evaluation framework for system *development*, as well as highlight a number of issues discovered in the process of applying the evaluation framework in practice.

2. Name Matching Evaluation Scenario

We will consider the following name matching scenario:

- A database contains records with different name variants.
- User enters a single name and wants to see all relevant records.
- In practice, a user sees a *ranked* list of name matches ordered by their similarity to the entered name

The scenario suggests the following evaluation protocol:

- Create a gold standard set of *keys*:
  - Query → True name matches in a DB.
- Run a name matching system to produce *output*:
  - Query → Ranked list of DB names
- Evaluate by comparing the system output to the keys and computing a *score*.

A diagram depicting a name matching system evaluation scenario is shown in Figure 1.

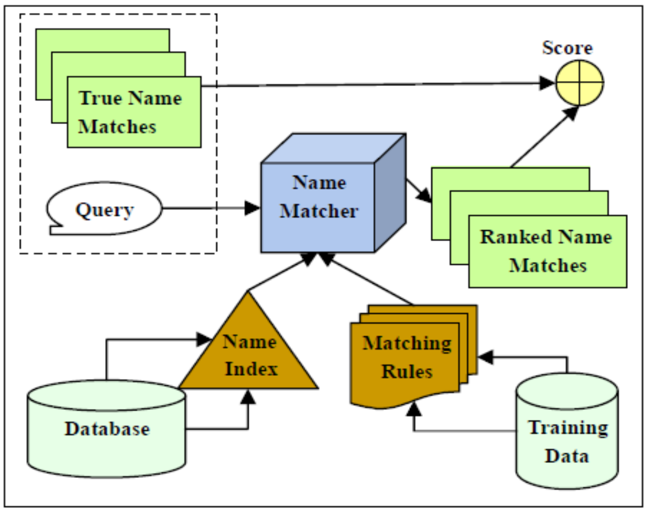


Figure 1: Name Matching System Evaluation

3. Name Matching Evaluation Metrics

Let us consider ramifications of using standard information retrieval evaluation metrics (van Rijsbergen 1979) for name matching.

3.1. Precision, Recall, and F-Measure

If a query name q has m true name variants $T_q = \{ t_1, t_2, \ldots, t_m \}$ in a DB, and a system returns n name variants $O_q = \{o_1, o_2, \ldots, o_n\}$, for the query q, then system recall R_q is (with respect to query q) defined as the percentage of true name variants that are returned by the system:

$$R_q = \frac{|T_q \cap O_q|}{|T_q|}$$

System precision P_q (with respect to query q) is defined as the percentage of output name variants that happen to be true:

$$P_q = \frac{|T_q \cap O_q|}{|O_q|}$$

Typically, a single number reflecting the overall system performance (with respect to query q) is expressed as the F-measure representing the harmonic mean of precision and recall:

$$F_q = \frac{2}{\frac{1}{P_q} + \frac{1}{R_q}}$$

While precision, recall, and F-measure are all established measures, they present a problem when developing name matching systems. A typical name matching system produces a ranked list of name matches allowing users to focus quickly on the more relevant matches. However, the above evaluation measures do not take ranking into account thereby sometimes giving the same evaluation scores to qualitatively different systems.
Let us consider an example of using precision, recall, and F-measure in comparing outputs of two name matching systems.

- Query: Andrzej K. Tarkowski
- Keys: Andrzej Tarkowski, Andrej Tarkovskij, Andrey Tarkovsky, Andrei Tarkovsky

System A		System B	
Name	**Score**	**Name**	**Score**
Andrzej Tarkowski	0.99	Andrzej Sapkowski	0.6
Andrej Tarkovskij	0.8	Andrzej Malkowski	0.6
Andrzej Sapkowski	0.6	Andrzej Tarkowski	0.6
Andrzej Malkowski	0.6	Andrej Tarkovskij	0.5

Table 1: System Comparison Example

Outputs of two systems, A and B, are shown in Table 1. While the systems return the same sets of names, the correct names are at the top of result list for system A, and at the bottom of the result list for system B. Thus, even though quantitatively precision, recall, and F-measure of the two systems are the same and equal to 0.5, users would prefer output produced by system A. Therefore, we seek a metric that would reflect not only system results, but also the results ranking.

3.2. Mean Average Precision (MAP)

If a system produces a ranked list of name variants $O_q = \{o_1, o_2, \ldots, o_n\}$, for a query q, we define precision at $k \leq n$, $P_q(k)$, to be the precision of the top k name variants $O_q(k) = \{o_1, o_2, \ldots, o_k\}$:

$$P_q(k) = \frac{|T_q \cap O_q(k)|}{|O_q(k)|}$$

Then, mean average precision (MAP) is defined as the average over all such $P(k)$ that o_k is a true name variant:

$$MAP_q = \frac{1}{m} \sum_{o_k \in T_q} P_q(k)$$

Mean average precision is a metric that is very sensitive to system output ranking. In the previous example,
MAP(System A) = (1+1)/4 = 0.5
MAP(System B) = (1/3+2/4)/4 = 0.21
The scores reflect the user intuition that output of system A is preferable to that of system B.

Numerous information retrieval evaluations found that the mean average precision is a robust metric, and it correlates fairly well with user judgments of system utility (Buckley & Voorhees 2000).

4. Incomplete Keys

When we started using mean average precision as the metric for evaluating and tuning a name matching system, we discovered that sometimes it leads to exaggerated penalties due to inherent *incompleteness* of name matching keys.

In order to illustrate keys incompleteness, let us consider the following example.

- Query: Mohammed Shahid

Keys	Output
• Mohamed Shahid • Jawan Muhammad Hussain Shaheed	1. Mohamed Shahid 2. Mohd. Shahid 3. M. Shahid 4. Muhammad Hussain Shaheed 5. Tufail Muhammad Shaheed 6. Muhammad Mahfuz Shaheed 7. Sawar Mohammad Hussain Shaheed 8. Jawan Muhammad Hussain Shaheed 9. Sowar Mohammad Hussain Shaheed

Table 2: Incomplete Keys Example

MAP of the system: (1+2/8)/2 = 0.625.

Table 2 shows that keys contain just two names matches for the query, while system output has seven more matches, all of which appear as good as the key matches. One may ask why it is the case that the good name matches are not part of the keys.

The reason has to do with an inherent difficulty of key construction for any realistic name matching evaluation. The word *realistic* is crucial here, because it means that in order to faithfully reflect the real world, the underlying database has to contain hundreds of thousands or even millions of names. For such large databases of names, name matching key construction is very difficult, for it implies an exhaustive scan of all database names. Note that such a scan cannot be conducted with system's help, since it would bias the keys towards system output – and it is imperative that the keys be constructed independently of any name matching system. As a manual exhaustive examination of a large database is impossible, any independent manually constructed keys are inherently incomplete.

We note that one can somewhat sidestep the issue of incompleteness when conducting a name matching evaluation aimed at comparison of competing systems by employing a method of "pooling" system results for construction of system-dependent sets of keys. This is the approach used by TREC (Voorhees & Harman 2000) and adopted for name matching evaluation as well (Miller *et al.* 2008). Unfortunately, the approach pre-supposes fixed name matching systems and extensive human effort required for cleaning and aggregating name matching results from multiple systems. And if any of the systems changes, the pooling process has to be repeated. Therefore, such an approach is not appropriate for name matching system development where we need quick automatic feedback indicating system performance.

Constructing a realistic dataset of name matching keys is an open problem. In practice, such a dataset occasionally arises as a result of a database construction process. In section 5, we present one such example of a large dataset, where key incompleteness is manageable.

4.1. Mitigating Key Incompleteness in Evaluation: Recall at *K*

Let us consider a typical search scenario, where a user enters a query name and looks at a ranked list of top *K* results (e.g., *K=10*). We want the top *K* list to contain as many true matches as possible. We quantify the intuition by introducing another evaluation metric: recall at *K*.

If a system produces a ranked list of name variants $O_q = \{o_1, o_2, \ldots, o_n\}$, for a query *q*, we define recall at *k*, $R_q(k)$, to be the recall of the top *k* name variants $O_q(k) = \{o_1, o_2, \ldots, o_k\}$:

$$R_q(k) = \frac{|T_q \cap O_q(k)|}{\min(k, |T_q|)}$$

Note that the minimum in the denominator is necessary to avoid penalizing recall for queries having more than *K* true matches.

The recall at *K* metric is largely insensitive to key incompleteness. For example, for the "Mohammed Shahid" query, $R_q(10) = 1.0$, since both true name matches are contained in top 10 results output by the system.

Selecting *K* is important, and it should be driven by both the likely number of true matches per query, the degree of incompleteness expected in the keys, as well as underlying name matching application requirements. As mentioned before, *K* is usually selected to be greater than the number of true matches. If keys are (unrealistically) expected to be complete, then we can select *K* on the per-query basis, setting $K = |T_q|$. In this case, recall at K, precision at K, and the corresponding F-measure are equal: they represent the break-even point of the system.

In a realistic situation, the choice of *K* is usually guided by the application requirements: if a system user typically focuses on top 10 search results, then it is natural to select *K=10*.

In our experiments, we found that recall at *K* is a more robust metric; it seems to better reflect utility of a system, and we use it a primary evaluation metric for system development. We also find it helpful to look at the curve of recall at K for increasing values of K, and compare the curves for different systems configurations.

Similar to the original evaluation metrics from Section 3.1, recall at *K* suffers from its inability to explicitly incorporate ranking information into the score. Note that ranking information is used implicitly, because ranking mistakes may often lead to true matches not being present within top *K* results. Yet it is desirable to account for ranking mistakes explicitly as well, and we suggest using mean average precision as a secondary evaluation metric to complement recall at *K*.

In Section 5, we give examples of using the two metrics in development of our name matching system.

5. Applying Metrics

We applied the name matching evaluation methodology to development of our name matching system.

We created name matching keys using Wikipedia redirects (alternative names of Wikipedia pages) with post-manual cleanups. We collected more than 10,000 people name queries and corresponding name equivalence classes, and the underlying database of more than 300,000 people names

For example, Table 3 shows redirects for the name "Zainal Abidin".

Redirects
Ali Zain al Abidin
Ali Zain-ul-Abideen
Ali Zayn al Abidin
Ali Zayn al-Abidin
Ali b. Husayn
Ali bin Hussein
Ali ibn Husayn
Ali ibn Husein
Ali ibn Hussayn
Zain Al-Abidin
Zain al-Abideen
Zain al-Abidin
Zain ul Abideen
Zain-ul-Abideen
Zain-ul-Abideen ibn Husayn
Zainul Abedeen
Zainul Abideen
Zainul Abidin
Zainulabideen
Zayn al-'Ābidin
Zayn al-Abidin
Zaynul Abideen

Table 3: Wikipedia Redirects Example

Our evaluation methodology proved to be an invaluable tool in the system development process allowing us to focus on the most important aspects of the system. The set of Wikipedia keys were equally split in development and blind test sets, each containing more than 5,000 name queries. The development test set was used to tune the system parameters. The final scores on the blind test part of the set of Wikipedia keys are shown in Figure 2.

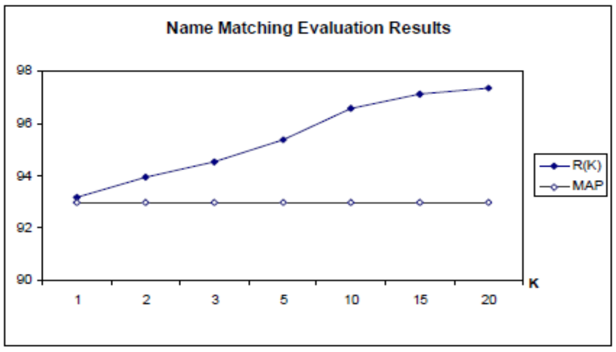


Figure 2: Evaluation Results

Recall at *K* increases with increasing the value of *K*, while MAP is independent of *K*. We tuned the system by seeing how system design decisions affect the recall at K curves and the value of MAP. A design decision was generally accepted if it raised both the *R(k)* curve and the value of MAP.

6. Conclusions

We presented a name matching evaluation methodology that is particularly suited for use in system development. We employed an implementation of the presented methodology, in conjuction with a set of name matching keys created from Wikipedia, in developing and tuning a state-of-the-art name matching system. An implementation of the proposed methodology is released and available upon request.

7. References

Buckley, C., Voorhees E. (2000) Evaluating evaluation measure stability. In N. Belkin, P. Ingwersen, M. Leong, editors. Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2000), Athens, Greece, 2000.

Miller K., Arehart, M., Ball, C., Polk, J., Rubenstein, A., Samuel, K., Schroeder, E., Vecchi, E., Wolf, C. (2008). An Infrastructure, Tools and Methodology for Evaluation of Multicultural Name Matching Systems, In N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odjik, S. Piperidis, D. Tapias, editors, Proceedings of the Sixth International Language Resources and Evaluation (LREC'08), Marrakech, Morocco, 2008, European Language Resources Association (ELRA).

van Rijsbergen, C. (1979). *Information Retrieval*, 2nd edition. Butterworths.

Voorhees, E., Harman, D. (2000). Overview of the Eighth Text REtrieval Conference (TREC-8), In D. Harman, editor, The Eighth Text Retrieval Conference (TREC-8), Gaithersburg, MD, USA, 2000, U.S. Government Printing Office, Washington D.C.

Resources for Named Entity Recognition and Resolution in News Wires

Rosa Stern[1,2] and Benoît Sagot[1]

1. Alpage, INRIA Paris-Rocquencourt & Université Paris 7, 30 rue du Château des Rentiers, 75013 Paris, France
2. Agence France-Presse – Medialab, 2 place de la Bourse, 75002 Paris, France
rosa.stern@afp.com, benoit.sagot@inria.fr

Abstract

In the applicative context of news wire enrichment with metadata, named entity recognition plays an important role, but requires to be followed by a resolution module that maps named entity mentions to entries in a reference database. In this paper, we describe NP, the named entity module embedded in the SxPipe shallow processing chain, that we used for extracting information from French news wires from the Agence France-Presse. We describe the construction of our reference database from freely available external resources, as well as our named entity detection, disambiguation and resolution modules. We also introduce a freely available and manually developped annotated corpus designed for the evaluation of named entity recognition and resolution tools, and provide evaluation figures for NP.

1. Overview

1.1. Introduction

Reference resolution is a challenge any named entity recognition system (henceforth NER) is confronted with as soon as its integration within the frame of a real application is in question. As studies like [Blume, 2005] have stated, reference resolution is obviously beneficial, necessary indeed, for an application involving NER in the prospect of exploiting information provided by named entities (henceforth NE) detected throughout data. Identification of certain text segments as NE phrases doesn't bring the whole information conveyed by NE usage without this segment — or mention — being linked to an extra-textual reference. Any particular application needing this association assumes the existence of such references, or more precisely of a reference resource to be the target of this linking.

After reviewing some considerations which are of importance in the development of an NER and resolution system, we present the application context of our research, which involves a dedicated reference base (section 2.). The creation and maintenance of the latter depends on the performance and functionalities of the system described in section 3., which addresses both entity recognition and reference resolution in a modular yet integrated fashion, applied to data in French. While the NER module (section 3.2.) involves some features which are peculiar to French and must be adapted in order to handle other languages, the resolution module presented in 3.3. is not language specific and is portable in the different steps of the system's evolution. Our system's development has lead to the creation of an evaluation resource, presented in section 5. along with results and future work outline.

1.2. Reference ambiguity

As done in the frame of ACE shared task [ACE08, 2008], NE mentions can be resolved to a unique reference through an identifier which disambiguate any linguistic metonymy or variation phenomena.

Entity resolution indeed consists of assigning a reference to phrases detected in texts as named entities mentions. In order to address this task within an NER system, the limits of classical entity classifications through static type categories have to be stressed (as in [Poibeau, 2005] or [Murgatroyd, 2008]). Such limits concern cases like metonymy ("I've read Proust and didn't like it"), precise oganization typing ("The UN will act on this violation"), entity inclusion in complex entities ("Kennedy Airport"). These cases illustrate the similarity between the polysemic behaviours of NEs and terms [Brun et al., 2009b]. The handling of entity reference in any NER system therefore implies the integration of extra-linguistic values of NEs, which is not necessarily obvious in their linguistic usage. Those characteristics of NE mentions make entity reference more subtle than a one-to-one matching between a detected phrase in texts and an identifier. In particular, the typing of NE mentions at the textual level must both guide the resolution process and avoid confusion. In the case of "Proust" in the above example, the fact that this particular mention doesn't refer to the person identified as the French writer *Marcel Proust* but, by metonymy, to his work raises the issue of the actual reference of this mention. Whether it should be resolved to the person *Marcel Proust* or not is a decision to be made in the context of the particular final application, but the linking of this mention to this reference must at least be detected and stated by the system. In the case of "Kennedy Airport", it is more obvious that the correct typing of this entity mention (type *facility*) should prevent any reference resolution to the person *J.F. Kennedy*. NE mentions found in texts can thus be treated as ambiguous occurences which an appropriate tool, such as described in section 3.3., can resolve to an identifier after a more general detection phase by taking into account several relevant elements given by the text about the extra-linguistic context of those occurences.

The other challenging aspect of entity reference, not without many connexions with the problem of polysemy, is the multiplicity of ways available to refer to a particular entity on a linguistic and textual level (i.e. a form of synonymy). Whether graphic or grammatical variants are in question (as “Mr Obama”, “B. Obama” or “He”, “The US president” for Barack Obama), such configurations raise the issue of an obvious matching between a mention of a NE and an identifier. Whereas the case of graphic variants are relatively easy to predict and handle within a surface analysis based tool, the grammatical anaphoras and coreferences demand to be resolved at a a deeper and more complex analysis level.

2. Application Context

2.1. Information Enrichment with Metadata

Alpage and the Agence France Presse (AFP) Medialab department have been involved in several projects dealing with information extraction from news wires and enrichment of AFP’s production. One of the main prospect of this research is to build a general and standardized reference base of metadata used in the production of news. By indexing news wires according to this resource, a structured and coherent production enrichment would be possible. This would then help for the improvment of various applications specific to a press agency, such as the filtering of the news production with customer-specific parameters or documentation tasks through information research in archived news.

The NER system which we present here has already been integrated to the SAPIENS platform, which is a prototype of news wires exploitation for information retrieval and knowledge extraction, on the particular topic of quotations. SAPIENS allows for a full quotations detection by matching them with their author, i.e. NEs mentioned in news wires. The user can therefore select some name among the list of detected entities and consult the quotations made by the corresponding person, all grouped together or in the context of the original news wire and of other entities related to them.

2.2. Workflow

This information enrichment with metadata has two aspects: it involves on the one hand the detection of NE as they are considered a decisive kind of metadata for the description of news content; on the other hand, not the entire set of metadata detected is considered relevant for a reference base whose main purpose is to reflect a state of knowledge inferable from the news production. As such, entities detected in news texts are always relevant to the description of the particular item they were found in, whereas they are not always relevant to the general and high-level reference base. NER in this context must therefore handle reference resolution at two levels. The detection tool itself has access to various databases, designed for NER and containing references; the matching of detected entities to references in those databases is then followed by a lookup in the reference base. At this point it must check the obtained reference against the reference base records and do one of these two actions: link the entity to its matching record if it exists, or propose the entity as a candidate to the reference base. The NER system thus updates its resources every time it operates this confrontation: an entity which is actually a record of the reference base should never be skipped, whereas a candidate entity rejected as a reference base record must not be reevaluated each time it occurs in texts . This information about entities reference must be taken into account by the NER system and passed on to its resources, which thus evolve along with their confrontation to new data. Reference resolution in our project thus happens at two levels: the level of the NER system resources, and the level of the reference base designed for the knowledge description of the AFP production. This modularity widely influences the type of resources used by the NER system, i.e. they have to include a set of entities references as large, relevant and exhaustive as possible in order to propose adequate candidates to the matching with the reference base. The choice and usage of those resources is described in section 3.1. It can be noted that among the different types of entities detected, some give more rise to reference resolution than others, i.e. *Persons* and *Organizations* show more peculiarities regarding resolution tasks as outlined in introduction than *Locations*.

In the particular case of a press agency, reference resolution takes on further complexities which have to be addressed, the first of which being the dynamic nature of the data. As news wires are produced every day (800 in French and a total of 5000 in the six languages in use at the AFP), the information they report and convey by definition introduces new references as new events happen, involving existant and new entities. The latter are either entities which have no previous record in references resources, or which do have one but were not considered as relevant entries for the reference base. Both configurations must be addressed, in the first case by proposing new references to handle unknown entities and in the second by promoting the entity status to the one of candidate for the reference base.

2.3. Reference Base

The building of the reference base is driven by a description of the data it should represent, i.e. a conceptual description of the knowledge reported and conveyed by the news production. This conceptual description takes the form of an ontology whose core is the relation network organizing entities. The entities classes outlined by this ontology correspond to a certain extent to a usual entities typology, mainly to a classification among *Person*, *Organization* (including companies, institutional organizations, sport teams...) and *Location* (including geopolitical entities as well as points of interests or facilities). This conceptual model for the reference base is also set to reflect the news themes already in use as metadata at the AFP and directly ensued by the IPTC taxonomy [1]. Politics, Economy, Culture or Sport are such themes and come along with a series

[1] `http://www.iptc.org/`

of subcategories used to describe news content. Those will have to be integrated in the network formed by the entities found in texts and populating the reference base.

The reference base in itself is currently in development. The phase consisting in matching detected entities against this base and in updating NER resources accordingly is therefore not fully realized yet in the running of our system. However, this preliminary usage of the system is not isolated from the overall application. It will indeed be used to build a first set of records to be integrated to the base, before next iterations where the lookup and candidacy phases take place.

The integration of NE recognition and resolution in this application therefore allows for the reference base population, as well as for its maintenance: the news production will be processed by the NER system combined with other specialized modules; each news item will then be indexed according to the data extracted by the system, depending on their mapping with the reference base.

3. NP: a system for NER and Entity Reference Disambiguation

Our NER and Entity Reference Disambiguation system is a part of SxPipe, a robust and modular surface processing chain for various languages and unrestricted text, used for shallow analysis or pre-preprocessing before parsing [Sagot and Boullier, 2005; Sagot and Boullier, 2008]. SxPipe is a freely-available[2] set of tools which performs (1) "named entities" recognition: pre-tokenization named entities (URLs, emails, dates, addresses, numbers…), (2) tokenization and segmentation in sentences, (3) token-based named entities (phrases in foreign languages…), (4) non-deterministic multi-word units detection and spelling error correction, and (5) lexicon-based patterns detection. The NE module within SxPipe, called NP (from the French *Noms Propres*, "Proper Nouns"), belongs to the 5th step.

NP is divided in two different steps described below. The first step is a non-deterministic detection module, developed in SxPipe's `dag2dag` framework [Sagot and Boullier, 2008]. This framework allows for defining context-free patterns and for using dedicated gazetteers, while remaining very efficient in terms of processing time. The second step disambiguates the ambiguous output of the first step and resolves the NEs that are retained, w.r.t. a NE database described below. These two steps need not be consecutive. The rationale behind this is that other modules that are applied between NE detection and NE disambiguation/normalization could achieve some disambiguation. For example, in the above-described SAPIENS system for quotation detection, the verbatim quotation detection module chooses the type *Person* when a NE is interpreted as the author of a verbatim quotation, even when it is competing with other types (e.g., *Marseille*, which is both a city and the last name of a mediatic historian and economist).[3]

Type	n# of entries	n# of variants
Person	263,035	883,242
Location	551,985	624,175
Organization	17,807	44,983
Work	27,222	59,851
Company	9,000	17,252
Product	3,648	6,350

Table 1: Quantitative data about the NE database underlying NP. Note that we have focused mostly on person, organization and location names.

Both modules rely on a large NE database extracted from several large-scale information repositories, mostly Wikipedia and `GeoNames`, as we shall now describe.

3.1. Resources and NE database building

Our NE database contains almost 900 000 entries and approx. 1,6 million NE denotation variants. It contains location, organization, person, company, product and work (booktitle, movie title…) names. More detailed quantitative information is shown in Table 1.

We extracted this database from two different sources: `GeoNames` for location names,[4] and the French Wikipedia for the other types of NE.[5] Each entry has a unique id, either a `GeoNames` id or an URL pointing to a Wikipedia article.

For location names, we filtered the `GeoNames` database using criteria defined according to the nature of the corpus. Because of the size of this database, we did not retain all entries and all aliases for each entry. Instead, we kept all entries concerning France and all entries corresponding to villages, towns and administrative regions in other coutries, provided their population is known to `GeoNames` and equals at least 200 people. Moreover, we discarded all location names with a non-empty language indication different than "French" or that contained non-French characters. For each retained location name, we store the `GeoNames` id, the `GeoNames` normalized name, and the latitude and longitude. We also compute a weight from the number of inhabitants when it is provided by `GeoNames`. This weight will be used during the NE disambiguation step. Moreover, this weight allows us to compute a reasonable scale level for use in the final interface when showing the location in *Google Maps*. In case of homonymy, we assign unique normalized forms by appending an index to the original normalized form of each

[2]`https://gforge.inria.fr/projects/lingwb/`, distributed under an LGPL license.

[3]In fact, our system is slightly more complex. In particular, it knows that the name of a capital city can be the author of a quotation, but should not be typed *Person* anyway. This is one of the most frequent cases of metonymy, described in introduction.

[4]Freely available at `http://www.geonames.org`

[5]A full dump can be downloaded freely at `http://download.wikimedia.org/frwiki/latest/frwiki-latest-pages-articles.xml.bz2`.

entry but the first one (e.g., there are 14 entries for locations named *Paris*, whose normalized forms are respectively *Paris*, *Paris (2)*,... *Paris (14)*).

For other kinds of NEs (persons, organizations, companies, products and brands, artworks), we extracted information from the (French) Wikipedia. The exploitation of Wikipedia for NE detection and resolution is not new, but has been proved efficient [Balasuriya et al., 2009]. We manually defined a mapping from a set of Wikipedia "categories" to one of the above-mentioned NE types. This allowed to type the title of each relevant Wikipedia article. Each typed article gives birth to an entity, whose normalized form is built from the title of the article, to which a disambiguation index may be appended, as for the case of locations. Apart from the title of the article, we extract other mention "variants" by two different means:

- we parse the first sentence of each article and automatically extract variants from it (e.g., *CIA* in addition to *Central Intelligence Agency*, or *Marie-Ségolène Royal* in addition to *Ségolène Royal*); we also extract a "definition" for the entity (in the case of Ségolène Royal, *femme politique française (22 septembre 1953, Dakar –)*).
- we retrieve all redirection pages in the Wikipedia and consider their titles as variants for the entity denoted by the target page.

In the case of person names, additional variants are computed. Indeed, the set of already gathered variants and a large-coverage lexicon of first names extracted from our general-purpose French lexicon allow us to segment person names into the first name, a possible middle name, the last name, and a gender if possible [6]. New variants are then computed, in particular the omission or abbreviation of first and middle names, as in *M.-S. Royal* or *Royal*.[7] As for locations, we assign a weight to all entities extracted from Wikipedia, that will be used during the NE disambiguation step. We compute this weight in a very simple way, based on the size (number of lines) of the whole Wikipedia article.

The output of this extraction process from `GeoNames` and Wikipedia is corrected and enriched by a blacklist and a whitelist of NEs, both manually drawn up. From the resulting NE database, we extract a gazetteer from all variants of each NE, associated with its type (and gender for person names when available).[8]

[6]This person name variant extraction heuristics is specific to names from languages such as French that write the given name before the family name, which is the case in most person name occurrences in our corpus. Other orders should be taken into account in further work.

[7]A candidate such as *Royal*, i.e. an isolated last name, is discarded during the disambiguation step unless it refers to an entity mentioned earlier in the same news item in a more extended form, e.g. *Ségolène Royal*

[8]As part of SxPipe, this database is freely available within the SxPipe distribution.

3.2. NE Recognition

A context-free grammar consisting of 130 rules has been developed for defining patterns based on this gazetteer, as well as on specific lists for identifying relevant contexts (e.g., *ville*, *village*, *localité*, i.e., *city*, *village*, *locality*; another example is a large list of first names, a list of possible titles such as *Dr.*, *Mme*, and others). Disambiguation heuristics have been activated,[9] in order to make the amount of ambiguities added by this NE module as low as possible, although not null. Therefore, the output of the NE Recognizer is a DAG (Directed Acyclic Graph) in which each possible NE span and type combination is represented by a distinct path. Note that recognition involves typing. Therefore, in case of NE type ambiguity, this will be treated as NE recognition ambiguity, and solved by the NE disambiguation and normalization step.

3.3. NE Disambiguation and Resolution

The aim of the NE disambiguation and normalization module is to chose at most one NE reading for each sequence of tokens recognized in the previous step. A *reading* is defined as the combination of a path in the input DAG and an entry in the NE database (i.e., one path in the input DAG may correspond to several readings). Unlike [Pilz and Paaß, 2009] and others, our disambiguation module relies on heuristics based on quantitative and qualitative information, but not on machine learning techniques.

First, we define the *salience level* of an entity as follows. Within a given document, each mention of an entity increments its salience level by its weight. Moreover, we define for each document a geographic context (country and city) by storing all countries and cities mentioned in the document. Each mention of a location entity increments its salience level by an additional (small) weight if it is consistent with the geographic context. On the other hand, we divide by 2 the salience level of all entities each time we move from a document to the next one (here, documents are news wire items).[10]

The strategy we use to select a unique reading, or a limited number of competing readings, can be summed up as follows. NE readings may be tagged as "dubious" or "normal", according to various heuristics that involve the type, the salience level and the surface form of the mention of the entity, as well as its left context. For example, a person name is considered dubious if it is a single token and if it is not the last name of an entity that has been previously detected. If dubious and normal readings share at least one token, dubious readings are discarded.

Among remaining NE readings, we reduce the ambiguity as follows. Among all readings corresponding to the

[9]E.g., longest match heuristics in some cases, preferences between pattern-based and gazetteer-based detection, and others.

[10]This division by 2 is merely a rule-of-thumb choice. In further work, we intend to conduct a series of experiments in order to optimize the salience dynamics w.r.t. performance levels.

Person	Total	Known	Unknown
References	223	111	112
Mentions	672	252	172
Location	Total	Known	Unknown
References	261	217	44
Mentions	672	613	59
Organization	Total	Known	Unknown
References	196	101	95
Mentions	463	316	147

Table 2: Corpus figures

same path, we keep only the reading that corresponds to the entity with the highest salience level.

Finally, in order to retain only one reading, we apply a simple longest-match left-to-right heuristics. However, ambiguity can be preserved in some cases, e.g., if a manual validation step follows the automatic processing.

4. Creation of a Reference Corpus for NER in French

In order to evaluate our NER and resolution system, a subset of news made available to us by the AFP has been selected and manually annotated. This annotation has the form of inline XML tagging and includes both mention-level and reference-level features: span boundaries and type on the one hand, and unique identifier matching a record in the reference base on the other. We aim indeed at addressing NER and reference resolution in an integrated way, which reasonably entails a unique evaluation resource suitable for evaluating the performance of both modules as well as of their interaction (in a comparable way to what is described in [Möller et al., 2004]).

This set of news wires is made up of 100 items with an average of 300 words each. Table 2 shows the distribution over NE types and mentions with known and unknwon refrences.It is freely available within the SxPipe distribution.

The NE types which we included in the annotation are similar to the ones selected in most dedicated conference shared tasks; they are so far limited to *Person*, *Organization* and *Location*. The identifier is a copy of the record identifier in the NER resources (section 3.3.). If no matching record exists, the canonical form is then stated and an attribute indicating this absence of record is added to the mention tag. The NE mentions do not include tokens which are not part of the name itself, such as titles for person names (*Dr.* or *Mr.*).

Annotation examples:

- `Le président <Person name="Barack Obama">Barack Obama</Person> a approuvé un accord`

- `grippe porcine au <Location name="Canada (2)">Canada</Location> a été révisé`

- `<Person name="Mohammed Abdullah Warsame" ref="unknown">Mohammed Abdullah Warsame</Person>, 35 ans, habitant`

5. Evaluation

Development and Test Data The gold corpus was divided in two parts of equivalent size: a development set and a test set. This division between development and test data ensures that the accuracy of the system is not artificially improved by the knowledge of the whole data. One out of every two items forms the test set in order to avoid systematic biaises caused by specific NEs which occur more frequently in a specific part of the corpus.

Metrics As for the metrics applied, we rely on a classical F-score obtained by the harmonic mean of precision and recall. However we calculated more than one F-score, by considering different ways of defining the intersection between the number of entities retrieved by the system and the number of relevant entities occuring in the evaluation corpus.

At the level of NER, it seems indeed reasonable to consider as fully correct any detection of entity mention along with its exact boundaries (correctness of span) and type. This is for example the requirement of the Conll 2003 shared task [Sang and Meulder, 2003] and the scoring is then based on it. Our choice thus depart from scoring methods such as the one of the Message Understanding Conference (MUC) framework [Grishman and Sundheim, 1996] which also allows for partial credit in cases of partial span or wrong type detection. One could also consider that, depending on the application context, one feature or another is of more importance. For instance, extracting entities with incorrect spans or failing to get a maximal precision generates a noise which can be highly unconvenient with regards to the user experience and expectations. Precision can thus be favoured in the calculation of the F-score by giving it more weight than to recall. This focuses efforts on the improvement of precision, even if it means a drop of recall performance, if this is the result considered as the best suited for an application. The metrics used in MUC include this kind of considerations and produce several measures depending on the system feature which is put forward.

When including the performance of reference resolution within the evaluation, the retrieved-relevant intersection must consider as correct matches only the ones which show the right reference as one of their feature. Fully correct matches at the reference resolution level are thus the ones which show span, type and reference correction.

In practice we aim at obtaining three evaluation levels. First, at the recognition level, the accuracy of the system is scored according to the mentions whose span and type are correctly detected. Then we measure the ability of the system to detect that a mention should be matched against a reference record (as opposed to mentions that do not correspond to an entity present in the database). Last, we score

Sub-task	Prec.	Rec.	F-sc.
Detection (span & type are correct?)	0.81	0.77	0.79
Reference detection (entity known?) (among correct span & type)	0.97	0.99	0.98
Reference resolution (which entity?) (among correct span & type & known)	0.91	–	–

Table 3: Evaluation results for the three following subtasks on French news wires: NE recognition, NE reference detection, NE reference resolution.

the accuracy of the resolution step. The set of cases considered for this last measure is therefore the intersection of correctly recognized entities mentions with entities mentions linked to a reference.

Results Table 3 shows the results for the three evaluation levels of our system running on the test gold set. As can be seen, the step that exhibits the lowest F-score is the detection step. The detailed F-scores for Person names, Location names and Organization names are respectively 0.80, 0.85 and 0.68. This can be compared to results of [Brun et al., 2009a], which respectively reach 0.79, 0.76 and 0.65 [Jacquet, p.c.]. The two other steps are difficult to compare to other work, especially for French. However, during the development of NP, we have seen how sensitive the results are to the quality of the NP reference database extraction, as well as to the heuristics used during the resolution step. Therefore, we consider that there is still room for significant improvements both at the detection and at the resolution steps. This includes the conjunction of our rule- and resource-based techniques with machine-learning methods.

6. Future Work

A more finalized reference base, structured by an underlying ontology, shall be used in further versions of NP. We also intend to improve NP's ability to detect new potential entities references appearing in texts. The types of entities handled by the system should be extended beyond the limitations stated in 4. by integrating more entity types, in an extent comparable to the categories defined for the Conll 2003 shared task, in particuliar the *Miscellaneous* category. This integration involves carrying out more annotation and double validation in order for the system to benefit from a complete evaluation resource. More generally, the multilingual production of the AFP should give rise to the development of resources and system adaptation for other languages than French.

Acknowledgments

This work was supported in part by the Scribo project funded by the French "Pôle de compétitivité" System@tic ("Fonds Unique Interministériel") and by the EDyLex project funded by the French National Research Agency (grant number ANR-09-CORD-008).

7. References

ACE08, 2008. Automatic content extraction 2008 evaluation plan.

Balasuriya, Dominic, Nicky Ringland, Joel Nothman, Tara Murphy, and James R. Curran, 2009. Named entity recognition in wikipedia. In *People's Web '09: Proceedings of the 2009 Workshop on The People's Web Meets NLP*. Morristown, NJ, USA: Association for Computational Linguistics.

Blume, Matthias, 2005. Automatic entity disambiguation: Benefits to ner, relation extraction, link analysis, and inference. *International Conference on Intelligence Analysis*.

Brun, C., N. Dessaigne, M. Ehrmann, B. Gaillard, S. Guillemin-Lanne, G Jacquet, A. Kaplan, M. Kucharski, C. Martineau, A. Migeotte, T. Nakamura, and S. Voyatzi, 2009a. Une expérience de fusion pour l'annotation d'entités nommées. In *Proceedings of TALN2009*.

Brun, Caroline, Maud Ehrmann, and Guillaume Jacquet, 2009b. A hybrid system for named entity metonymy resolution. In *LNCS 5603, Selected papers from the 3rd Language and Technology Conference (LTC 2007)*. Poznan, Poland: Springer-Verlag.

Grishman, Ralph and Beth Sundheim, 1996. Message understanding conference-6: a brief history. In *Proceedings of the 16th conference on Computational linguistics (CoLing'96)*. Copenhagen, Denmark.

Murgatroyd, David, 2008. Some linguistic considerations of entity resolution and retrieval. In *Proceedings of LREC 2008 Workshop on Resources and Evaluation for Identity Matching, Entity Resolution and Entity Management*.

Möller, Knud, Alexander Schutz, and Stefan Decker, 2004. Towards an integrated corpus for the evaluation of named entity recognition and object consolidation. In *Proceedings of the SemAnnot Workshop at ISWC2004*.

Pilz, Anja and Gerhard Paaß, 2009. Named entity resolution using automatically extracted semantic information. In *Proceedings of LWA 2009*.

Poibeau, Thierry, 2005. Sur le statut référentiel des entités nommées. *CoRR*, abs/cs/0510020.

Sagot, Benoît and Pierre Boullier, 2005. From raw corpus to word lattices: robust pre-parsing processing with SxPipe. *Archives of Control Sciences, special issue on Language and Technology*, 15(4):653–662.

Sagot, Benoît and Pierre Boullier, 2008. SxPipe 2 : architecture pour le traitement présyntaxique de corpus bruts. *Traitement Automatique des Langues (T.A.L.)*, 49(2):155–188.

Sang, Erik F. Tjong Kim and Fien De Meulder, 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2003*.