WiNER: A Wikipedia Annotated Corpus for Named Entity Recognition

Abbas Ghaddar
RALI-DIRO
Université de Montréal
Montréal, Canada
abbas.ghaddar@umontreal.ca

Philippe Langlais
RALI-DIRO
Université de Montréal
Montréal, Canada
felipe@iro.umontreal.ca

Abstract

We revisit the idea of mining Wikipedia in order to generate named-entity annotations. We propose a new methodology that we applied to the English Wikipedia to build WiNER, a large, high quality, annotated corpus. We evaluate its usefulness on 6 NER tasks, comparing 4 popular state-of-the art approaches. We show that LSTM-CRF is the approach that benefits the most from our corpus. We report impressive gains with this model when using a small portion of WiNER on top of the CONLL training material. Last, we propose a simple but efficient method for exploiting the full range of WiNER, leading to further improvements.

1 Introduction

Named-Entity Recognition (NER) is the task of identifying textual mentions and classifying them into a predefined set of types. It is an important pre-processing step in NLP and Information Extraction. Various approaches have been proposed to tackle the task, including conditional random fields (Finkel et al., 2005), perceptrons (Ratinov and Roth, 2009), and neural network approaches (Collobert et al., 2011; Lample et al., 2016; Chiu and Nichols, 2016).

One issue with NER is the small amount of annotated data available for training, and their limited scope (see Section 4.1). Furthermore, some studies (Onal and Karagoz, 2015; Augenstein et al., 2017) have demonstrated that named-entity systems trained on news-wire data perform poorly when tested on other text genres. This motivated some researchers to create a named-entity labelled corpus from Wikipedia. This was notably attempted by Nothman et al. (2008) and more recently revisited by Al-Rfou et al. (2015) in a multilingual context. Both studies leverage the link structure of Wikipedia to generate named-entity annotations. Because only a tiny portion of texts in Wikipedia are anchored, some strategies are typically needed to infer more annotations (Ghaddar and Langlais, 2016b). Such a process typically yields a noisy corpus for which filtering is required.

In this paper, we revisit the idea of automatically extracting named-entity annotations out of Wikipedia. Similarly to the aforementioned works, we gather anchored strings in a page as well as their type according to Freebase (Bollacker et al., 2008) but, more importantly, we also generate annotations for texts not anchored in Wikipedia. We do this by considering coreference mentions of anchored strings as candidate annotations, and by exploiting the out-link structure of Wikipedia. We applied our methodology on a 2013 English Wikipedia dump, leading to a large annotated corpus called WiNER, which contains more annotations than similar corpora and, as we shall see, is more useful for training NER systems.

We discuss related work in Section 2 and present the methodology we used to automatically extract annotations from Wikipedia in Section 3. The remainder of the article describes the experiment we conducted to measure the impact of WiNER for training NER systems. We describe the datasets and the different NER systems we trained in Section 4. We report the experiments we conducted in Section 5. We propose a simple but efficient two stage strategy we designed in order to benefit the full WiNER corpus in Section 6. We report error analysis in Section 7 and conclude in Section 8.

Proceedings of the The 8th International Joint Conference on Natural Language Processing, pages 413–422,
Taipei, Taiwan, November 27 – December 1, 2017

2 Related Work

Turning Wikipedia into a corpus of named entities annotated with types is a task that received some attention in a monolingual setting (Toral and Munoz, 2006; Nothman et al., 2008), as well as in a multilingual one (Richman and Schone, 2004; Al-Rfou et al., 2015).

In (Nothman et al., 2008) the authors describe an approach that exploits links between articles in Wikipedia in order to detect entity mentions. They describe a pipeline able to detect their types (ORG, PER, LOC, MISC), making use of handcrafted rules specific to Wikipedia, and a bootstrapping approach for identifying a subset of Wikipedia articles where the type of the entity can be predicted with confidence. Since anchored strings in Wikipedia lack coverage (in part because Wikipedia rules recommend that only the first mention of a given concept be anchored in a page), the authors also describe heuristics based on redirects to identify more named-entity mentions. They tested several variants of their corpus on three NER benchmarks and showed that systems trained on Wikipedia data may perform better than domain-specific systems in an out-domain setting.

Al-Rfou et al. (2015), follow a similar path albeit in a multilingual setting. They use Freebase to identify categories (PER, LOC, ORG), and trained a neural network on the annotations extracted. In order to deal with non-anchored mentions in Wikipedia, they propose a first-order coreference resolution algorithm where they link mentions in a text using exact string matching (thus *Obama* will be linked to the concept *Barack Obama* and labelled PER). They still had to perform some sentence selection, based on an oversampling strategy, in order to construct a subset of the original training data.

Our work revisits the idea developed in these two studies. Our main contribution consists in dealing specifically with non anchored strings in Wikipedia pages. We do this by analyzing the outlink structure in Wikipedia, coupled to the information of all the surface forms that have been used in a Wikipedia article to mention the main concept being described by this article. This process, detailed in the next section, leads to a much larger set of annotations, whose quality obviates the need for ad-hoc filtering or oversampling strategies.

3 WiNER

We applied the pipeline described hereafter to a dump of English Wikipedia from 2013, and obtained WiNER, a resource built out of 3.2M Wikipedia articles, comprising more than 1.3G tokens accounting for 54M sentences, 41M of which contain at least one named-entity annotation. We generated a total of 106M annotations (an average of 2 entities per sentence).

3.1 Annotation Pipeline

The pipeline used to extract named-entity annotations from Wikipedia is illustrated in Figure 1, for an excerpt of the Wikipedia article *Chilly_Gonzales*, hereafter named the target article. Similarly to (Nothman et al., 2008; Al-Rfou et al., 2015), the anchored strings of out-links in the target article are elected mentions of named entities. For instance, we identify *Warner Bros. Records* and *Paris* as mentions in our target article. In general, a Wikipedia article has an equivalent page in Freebase. We remove mentions that do not have such a page. This way, we filter out anchored strings that are not named entities (such as *List of Presidents of the United States*). We associate a category with each mention by a simple strategy, similar to (Al-Rfou et al., 2015), which consists in mapping Freebase attributes to entity types. For instance, we map `organization/organization`, `location/location` and `people/person` attributes to ORG, LOC and PER, respectively. If an entry does not belong to any of the previous classes, we tag it as MISC.

Because the number of anchored strings in Wikipedia is rather small — less than 3% of the text tokens according to (Al-Rfou et al., 2015) — we propose to leverage: (1) the out-link structure of Wikipedia, (2) the information of all the surface strings used to describe the main concept of a Wikipedia article. For the latter, we rely on the resource[1] described in (Ghaddar and Langlais, 2016a) that lists, for all the articles in Wikipedia (those that have a Freebase counterpart), all the text mentions that are coreferring to the main concept of an article. For instance, for the article *Chilly_Gonzales*, the resource lists proper names (e.g. Gonzales, Beck), nominal (e.g. the per-

[1] `http://rali.iro.umontreal.ca/rali/en/wikipedia-main-concept`

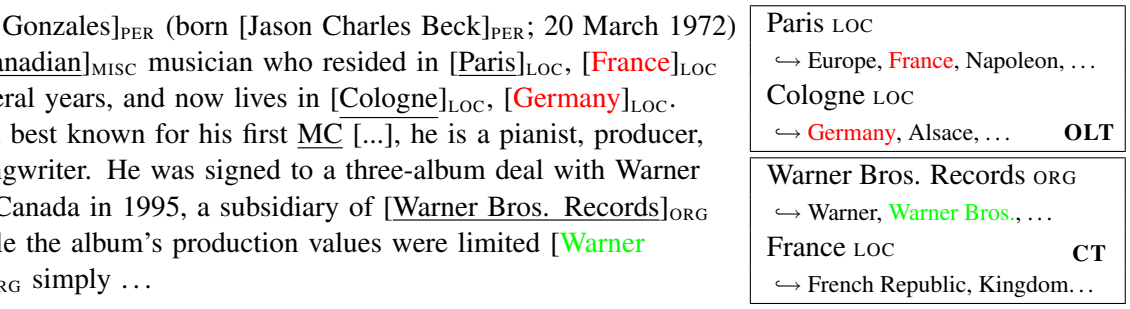

Figure 1: Illustration of the process with which we gather annotations into WiNER for the target page `https://en.wikipedia.org/wiki/Chilly_Gonzales`. Bracketed segments are the annotations, underlined text are anchored strings in the corresponding Wikipedia page. OLT represents the out-link table (which is compiled from the Wikipedia out-link graph structure), and CT represents the coreference table we gathered from the resource.

former) and pronominal (e.g. he) mentions that refer to Chilly Gonzales. From this resource, we consider proper name mentions, along with their Freebase type.

Our strategy for collecting extra annotations is a 3-step process, where:

1. We consider direct out-links of the target article. We search in its text the titles of the articles we reach that way. We also search for their coreferences as listed in the aforementioned resource. For instance, we search (exact match) *Warner Bros. Records* and its coreferences (e.g. *Warner*, *Warner Bros.*) in the target article. Each match is labelled with the type associated (in Freebase) with the out-linked article (in our example, ORG).

2. We follow out-links of out-links, and search in the target article (by an exact string match) the titles of the articles reached. For instance, we search for the strings *Europe, France, Napoleon*, as well as other article titles from the out-link list of the article *Paris*. The matched strings are elected named entities and are labeled with their Freebase type.

3. For the titles matched at step 2, we also match their coreferent mentions. For instance, because we matched *France*, we also search its coreferences as listed in the coreference table (CT).

During this process, some collisions may occur. We solve the issue of overlapping annotations by applying the steps exactly in the order presented above. Our steps have been ordered in such a way that the earlier the step, the more confidence we have in the strings matched at that step. It may also happen that two out-link articles contain the same mention (for instance *Washington_State* and *George_Washington* both contain the mention *Washington*), in which case we annotate this ambiguous mention with the type of the closest[2] unambiguous mention.

Step 1 of our pipeline raises the coverage[3] from less than 3% to 9.5%, while step 2 and 3 increase it to 11.3% and 15% respectively. This is actually very close to the coverage of the manually annotated CONLL-2003 dataset, which is 17%. Considering that we do not apply any specific filtering, as is done for instance in (Nothman et al., 2008), our corpus contains many more annotations than existing Wikipedia-based named-entity annotated corpora.

3.2 Manual Evaluation

We assessed the annotation quality of a random subset of 1000 mentions. While we measure an accuracy of 92% for mentions detected during step 1, the accuracy decreases to 88% and 77% during step 2 and 3 respectively. We identified two main sources for errors in the coreferent mentions detection procedure. One source of error comes from the resource used to identify the mentions of the main concept. We measured in a previous work (Ghaddar and Langlais, 2016a), that the process we rely on for this (a binary classifier) has an accuracy of 89%. Example (a) of Figure 2 illus-

[2]Before or after the named-entity.
[3]Ratio of annotated tokens.

trates such a mistake where the family name *Pope* is wrongly assumed coreferent to the brewery *Eldridge Pope*. We also found that our 3-step process and the disambiguation rule fails in 15% of the cases. Figure 2 illustrates an example where we erroneously recognize the mention *Toronto* (referring to the town) as a coreferent of the (non ambiguous mention) *Toronto FC*, simply because the latter is close to the former.

a) *[Eldridge Pope]*$_{\text{ORG}}$ was a traditional brewery.....Sixteen years later the *[Pope]*$_{\text{ORG}\star}$ brothers floated the business...

b) Montreal Impact's biggest rival is *[Toronto FC]*$_{\text{ORG}}$ because Canada's two largest cities have rivalries in and out of sport. Montreal and *[Toronto]*$_{\text{ORG}\star}$ professional soccer teams have competed against each other for over 40 years.

c) I didn't want to open up my *[Rolodex]*$_{\text{ORG}\star}$ and get everyone to sing for me.

Figure 2: Examples of errors in our annotation pipeline. Faulty annotations are marked with a star.

Table 1 shows the counts of token strings annotated with at least two types. For instance, there are 230k entities that are annotated in WiNER as PER and LOC. It is reassuring that different mentions with the same string are labelled differently. The cells on the diagonal indicate the number of mentions labelled with a given tag.

	PER	LOC	ORG	MISC
PER	28M	230k	80k	250k
LOC	-	29M	120k	190k
ORG	-	-	13M	206k
MISC	-	-	-	36M

Table 1: Number of times a text string (mention) is labelled with (at least) two types in WiNER. The cells on the diagonal indicate the number of annotations.

We further examined a random subset of 100 strings that were annotated differently (in different contexts) and found that 89% of the time, the correct type was identified. For instance, in example Figure 2c) — a sentence of the *Chilly_Gonzales* article — the mention *Rolodex* is labelled as ORG, while the correct type is MISC. Our pipeline fails to disambiguate the company from its product.

4 Protocol

4.1 Data Sets

We used a number of datasets in our experiments. For CONLL, MUC and ONTO, that are often used to benchmark NER, we used the test sets distributed in official splits. For the other test sets, that are typically smaller, we used the full dataset as a test material.

CONLL the CONLL-2003 NER Shared Task dataset (Tjong Kim Sang and De Meulder, 2003) is a well known collection of Reuters newswire articles that contains a large portion of sports news. It is annotated with four entity types (PER, LOC, ORG and MISC).

MUC the MUC-6 (Chinchor and Sundheim, 2003) dataset consists of newswire articles from the Wall Street Journal annotated with PER, LOC, ORG, as well as a number of temporal and numerical entities that we excluded from our evaluation for the sake of homogeneity.

ONTO the OntoNotes 5.0 dataset (Pradhan et al., 2012) includes texts from five different text genres: broadcast conversation (200k), broadcast news (200k), magazine (120k), newswire (625k), and web data (300k). This dataset is annotated with 18 fine grained NE categories. Following (Nothman, 2008), we applied the procedure for mapping annotations to the CONLL tag set. We used the CONLL 2012 (Pradhan et al., 2013) standard test set for evaluation.

WGOLD WikiGold (Balasuriya et al., 2009) is a set of Wikipedia articles (40k tokens) manually annotated with CONLL-2003 NE classes. The articles were randomly selected from a 2008 English dump and cover a number of topics.

WEB Ratinov and Roth (2009) annotated 20 web pages (8k tokens) on different topics with the CONLL-2003 tag set.

TWEET Ritter et al. (2011) annotated 2400 tweets (comprising 34k tokens) with 10 named-entity classes, which we mapped to the CONLL-2003 NE classes.

4.2 Metrics

Since we use many test sets in this work, we are confronted with a number of inconsistencies. One is the definition of the MISC class, which differs from a dataset to another, in addition to not being annotated in MUC. This led us to report token-level F1 score for 3 classes only (LOC, ORG and PER). We computed this metric with the `conlleval` script.[4]

We further report OD_{F1}, a score that measures how well a named-entity recognizer performs on out-domain material. We compute it by randomly sampling 500 sentences[5] for each out-domain test set, on which we measure the token-level F1. Sampling the same number of sentences per test set allows to weight each corpus equally. This process is repeated 10 times, and we report the average over those 10 folds. On average, the newly assembled test set contains 50k tokens and roughly 3.5k entity mentions. We excluded the CONLL-2003 test set from the computation since this corpus is in-domain[6] (see section 5.2).

4.3 Reference systems

We chose two feature-based models: the `StanfordNER` (Finkel et al., 2005) CRF classifier, and the perceptron-based `Illinois` NE Tagger (Ratinov and Roth, 2009). Those systems have been shown to yield good performance overall. Both systems use handcrafted features; the latter includes gazetteer features as well.

We also deployed two neural network systems: the one of (Collobert et al., 2011), as implemented by Attardi (2015), and the `LSTM-CRF` system of Lample et al. (2016). Both systems capitalize on representations learnt from large quantities of unlabeled text[7]. We use the default configuration for each system.

5 Evaluation of WiNER

5.1 Other Wikipedia-based corpora

We compare WiNER to existing Wikipedia-based annotated corpora. Nothman et al. (2008) released two versions of their corpus, WP2 and WP3, each containing 3.5 million tokens. Both versions enrich the annotations deduced from anchored strings in Wikipedia by identifying coreferences among NE mentions. They differ by the rules used to conduct coreference resolution. We randomly generated 10 equally-sized subsets of WiNER (of 3.5 million tokens each). On each subset, we trained the `Illinois` NER tagger and compared the performances obtained on the CONLL test set by the resulting models, compared to those trained on WP2 and WP3. Phrase-level F1 score are reported in Table 2. We also report the results published in (Al-Rfou et al., 2015) with the Polyglot corpus, which is unfortunately not available.

	with MISC		w/o MISC	
WP2	68.2		72.8	
WP3	68.3		72.9	
Polyglot	-		71.3	
WiNER	**71.2**	[70.3,71.6]	**74.5**	[73.4,75.2]

Table 2: Performance of the `Illinois` toolkit on CONLL, as a function of the Wikipedia-based training material used. The figures on the last line are averaged over the 10 subsets of WiNER we randomly sampled. Bracketed figures indicate the minimum and maximum values.

Using WiNER as a source of annotations systematically leads to better performance, which validates the approach we described in Section 3. Note that in order to generate WP2 and WP3, the authors applied filtering rules that are responsible for the loss of 60% of the annotations. Al-Rfou et al. (2015) also perform sentence selection. We have no such heuristics here, but we still observe a competitive performance. This is a satisfactory result considering that WiNER is much larger.

5.2 Cross-domain evaluation

In this experiment, we conduct a cross-domain evaluation of the reference systems described in Section 4.3 on the six different test sets presented in Section 4.1. Following a common trend in the field, we evaluate the performance of those systems when they are trained on the CONLL material. We also consider systems trained on CONLL plus a subset of WiNER. We report results obtained with a subset of randomly chosen sentences summing up to 3 million tokens, as well as a variant where we use as much as possible of the training material available in WiNER. Larger datasets

[4] `http://www.cnts.ua.ac.be/conll2000/chunking/conlleval.txt`

[5] The smallest test set has 617 sentences.

[6] Figures including this test set do not change drastically from what we observe hereafter.

[7] We use the pre-trained representations.

	CONLL	ONTO	MUC	TWEET	WEB	WGOLD	OD_{F1}
CRF							
CONLL	91.6	70.2	80.3	38.7	61.9	68.4	67.0
+WiNER(3M)	-	-	-	-	-	-	-
+WiNER(1M)	89.3 (-2.4)	71.8 (+1.7)	78.6 (-1.8)	49.2 (+10.5)	63.0 (+1.1)	69.1 (+0.8)	69.2(+2.2)
Illinois							
CONLL	**92.6**	71.9	84.1	44.9	57.0	71.4	68.3
+WiNER(3M)	85.5 (-6.9)	71.4 (-0.5)	76.2 (-7.9)	51.1 (+6.2)	**65.5 (+8.5)**	71.8 (+0.4)	69.5(+1.2)
+WiNER(30M)	82.0 (-10.6)	71.6 (-0.3)	75.6 (-8.5)	**52.2 (+7.3)**	63.3 (+6.3)	71.6 (+0.3)	69.0(+0.7)
Senna							
CONLL	90.3	68.8	73.2	36.7	58.6	70.0	64.3
+WiNER(3M)	86.6 (-3.7)	70.1 (+1.3)	73.9 (+0.7)	43.2 (+6.4)	62.6 (+4.0)	69.9 (-0.1)	67.0(+2.7)
+WiNER(7M)	86.8 (-3.5)	70.0 (+1.2)	72.9 (-0.4)	44.8 (+8.1)	61.5 (+2.9)	69.3 (-0.7)	66.2(+1.9)
LSTM-CRF							
CONLL	92.3	71.3	76.6	36.7	57.4	68.0	65.0
+WiNER(3M)	91.5 (-0.8)	74.7 (+3.4)	**84.7 (+8.1)**	48.1 (+11.4)	62.7 (+5.2)	73.2 (+5.2)	72.0 (+7.0)
+WiNER(5M)	91.1 (-1.2)	**76.6 (+5.3)**	84.0 (+7.4)	48.4 (+11.7)	64.4 (+7.0)	**74.3 (+6.4)**	**73.0 (+8.0)**

Table 3: Cross-domain evaluation of NER systems trained on different mixes of CONLL and WiNER. Figures are token-level F1 score on 3 classes, while figures in parentheses indicate absolute gains over the configuration using only the CONLL training material. Bold figures highlight column-wise best results.

were created by randomly appending material to smaller ones. Datasets were chosen once (no cross-validation, as that would have required too much time for some models). Moreover, for the comparison to be meaningful, each model was trained on the same 3M dataset. The results are reported in Table 3.

First, we observe the best overall performance with the LSTM-CRF system (73% OD_{F1}), the second best system being a variant of the `Illinois` system (69.5% OD_{F1}). We also observe that the former system is the one that benefits the most from WiNER (an absolute gain of 8% in OD_{F1}). This may be attributed to the fact that this model can explore the context on both sides of a word with (at least in theory) no limit on the context size considered. Still, it is outperformed by the `Illinois` system on the WEB and the TWEET test sets. Arguably, those two test sets have a NE distribution which differs greatly from the training material.

Second, on the CONLL setting, our results are satisfyingly similar to those reported in (Ratinov and Roth, 2009) and (Lample et al., 2016). The former reports 91.06 phrasal-level F1 score on 4 classes, while our score is 90.8 .The latter reports an F1 score of 90.94 while we have 90.76. The best results reported far on the CONLL setting are those of (Chiu and Nichols, 2016) with a BiLSTM-CNN model, and a phrasal-level F1 score of 91.62 on 4 classes. So while the models we tested are slightly behind on CONLL, they definitely are competitive. For other tasks, the comparison with other studies is difficult since the performance is typically reported with the full tagset.

Third, the best performances are obtained by configurations that use WiNER, with the exception of CONLL. That this does not carry over to CONLL confirms the observations made by several authors (Finkel et al., 2005; Al-Rfou et al., 2015), who highlight the specificity of CONLL's annotation guidelines as well as the very nature of the annotated text, where sport teams are overrepresented. These teams add to the confusion because they are often referred to with a city name. We observe that, on CONLL, the `LSTM-CRF` model is the one that registers the lowest drop in performance. The drop is also modest for the CRF model. The WiNER's impact is particularly observable on TWEET (an absolute gain of 8.8 points) and WEB (a gain of 5.5), again two very different test sets. This suggests that WiNER helps models to generalize.

Last, we observe that systems differ in their ability to exploit large training sets. For the two feature-based models we tested, the bottleneck is memory. We did train models with less features, but with a significantly lower performance. With the CRF model, we could only digest a subset of WiNER of 1 million tokens, while `Illinois` could handle 30 times more. As far as neural network systems are concerned, the is-

sue is training time. On the computer we used for this work — a Linux cluster equipped with a GPU — training `Senna` and `LSTM-CRF` required over a month each for 7 and 5 millions WiNER tokens respectively. This prevents us from measuring the benefit of the complete WiNER resource.

6 Scaling up to WiNER

6.1 Our 2-stage approach

Because we were not able to employ the full WiNER corpus with the NER systems mentioned above, we resorted to a simple method to leverage all the annotations available in the corpus. It consists in decoupling the segmentation of NEs in a sentence — we leave this to a reference NER system — from their labelling, for which we train a local classifier based on contextual features computed from WiNER. Decoupling the two decision processes is not exactly satisfying, but allows us to scale very efficiently to the full size of WiNER, our main motivation here.

6.1.1 Contextual representations

Our classifier exploits a small number of features computed from two representations of WiNER. In one of them, each named-entity is bounded by a beginning and end token tags — both encoding its type — as illustrated on line MIX of Figure 3. In the second representation, the words of the named-entity are replaced with its type, as illustrated on line CONT. The former representation encodes information from both the context and the the words of the segment we wish to label while the second one only encodes the context of a segment.

WiNER [Gonzales]$_{\text{PER}}$ will be featured on [Daft Punk]$_{\text{MISC}}$.

MIX ⟨B-PER⟩ Gonzales ⟨L-PER⟩ will be featured on ⟨B-MISC⟩ Daft Punk ⟨L-MISC⟩

CONT ⟨PER⟩ will be featured on ⟨MISC⟩ .

Figure 3: Two representations of WiNER's annotation used for feature extraction.

With each representation, we train a 6-gram backoff language model using `kenLM` (Heafield et al., 2013). For the MIX one, we also train word embeddings of dimension 50 using `Glove` (Pennington et al., 2014).[8] Thus, we have the embeddings of plain words, as well as those of token tags. The language and embedding models are used to provide features to our classifier.

6.1.2 Features

Given a sentence and its hypothesized segmentation into named-entities (as provided by another NER system), we compute with the Viterbi algorithm the sequence of token tags that leads to the smallest perplexity according to each language model. Given this sequence, we modify the tagging of each segment in turn, leading to a total of 4 perplexity values per segment and per language model. We normalize those perplexity values so as to interpret them as probabilities. Table 4 shows the probability given by both language models to the segment *Gonzales* of the sentence of our running example. We observe that both models agree that the segment should be labelled PER. We also generate features thanks to the embedding model. This time, however, this is done without considering the context: we represent a segment as the sum of the representation of its words. We then compute the cosine similarity between this segment representation and that of each of the 4 possible tag pairs (the sum of the representation of the begin and end tags); leading to 4 similarity scores per segment. Those similarities are reported on line EMB in Table 4.

	LOC	MISC	ORG	PER
CONT	0.11	0.35	0.06	**0.48**
MIX	0.26	0.19	0.18	**0.37**
EMB	0.39	0.23	0.258	**0.46**

Table 4: Features for the segment *Gonzales* in the sentence *Gonzales will be featured on Daft Punk*.

To these 4 scores provided by each model, we add 16 binary features that encode the rank of each token tag according to one model (does ⟨tag⟩ have rank ⟨i⟩ ?). We also compute the score difference given by a model to any two possible tag pairs, leading to 6 more scores. Since we have 3 models, we end up with 78 features.

6.1.3 Training

We use `scikit-learn` (Pedregosa et al., 2011) to train a Random Forest classifier[9] on the 29k mentions of the CONLL training data. We

[8] We used a window size of 5 in this work.

[9] We tried other algorithms provided by the platform with less success.

adopted this training material to ensure a fair comparison with other systems that are typically trained on this dataset. Another possibility would be to split WiNER into two parts, one for computing features, and the other for training the classifier. We leave this investigation as future work. Because of the small feature set we have, training such a classifier is very fast.

6.2 Results

We measure the usefulness of the complete WiNER resource by varying the size of the training material of both language models and word embeddings, from 5M tokens (the maximum size the LSTM-CRF mode could process) to the full WiNER resource size.

	CO	ON	MU	TW	WE	WG	OD_{F1}
5M	84.3	72.0	78.7	39.8	61.9	70.2	68.1
50M	86.8	75.6	82.3	44.9	64.7	73.8	71.7
500M	88.9	76.2	84.8	45.8	**66.6**	75.5	74.1
All	90.5	**76.9**	**85.9**	**46.6**	65.3	**77.0**	**74.7**

Table 5: Influence of the portion of WiNER used in our 2-stage approach for the CONLL test set, using the segmentation produced by LSTM-CRF+WiNER(5M). These results have to be contrasted with the last line of Table 3.

To this end, we provide the performance of our 2-stage approach on CONLL, using the segmentation output by LSTM-CRF+WiNER(5M) [10]. Results are reported in Table 5. As expected, we observe that computing features on the same WiNER(5M) dataset exploited by LSTM-CRF leads to a notable loss overall (OD_{F1} of 68.1 versus 73.0), while still outperforming LSTM-CRF trained on CONLL only (OD_{F1} of 65.0). More interestingly, we observe that for all test sets, using more of WiNER leads to better performance, even if a plateau effect emerges. Our approach does improve systematically across all test sets by considering 100 times more WiNER data than what LSTM-CRF can handle in our case. Using all of WiNER leads to an OD_{F1} score of 74.7, an increase of 1.7 absolute points over LSTM-CRF+WiNER(5M).

Table 6 reports the improvements in OD_{F1} of our 2-stage approach (RF), which uses all of the WiNER material and the segmentation produced by several native systems. Applying our 2-stage approach systematically improves the performance of the native configuration. Gains are larger for native configurations that cannot exploit a large quantity of WiNER. We also observe that the 2-stage approach delivers roughly the same level of performance ($OD_{F1} \simeq 74$) when using the segmentation produced by the Illinois or the LSTM-CRF systems.

[10] The best configuration according to Table 3.

	Native	**RF**
CRF		
CONLL	67.0	73.6 (+6.6)
+WiNER(3M)	-	-
+WiNER(1M)	69.2	73.0 (+2.8)
Illinois		
CONLL	68.3	74.4 (+6.1)
+WiNER(3M)	69.5	74.2 (+4.7)
+WiNER(30M)	69.0	74.3 (+4.3)
Senna		
CONLL	64.3	70.1 (+5.8)
+WiNER(3M)	67.0	70.8 (+3.8)
+WiNER(7M)	66.2	72.0 (+5.8)
LSTM-CRF		
CONLL	65.0	69.7 (+4.7)
+WiNER(3M)	72.0	**74.8 (+2.8)**
+WiNER(5M)	73.0	74.7 (+1.7)

Table 6: OD_{F1} score of native configurations, and of our two-stage approach (RF) which exploits the full WiNER corpus. Figures in parenthesis indicate absolute gains over the native configuration.

7 Error Analysis

Table 7 indicates the number of disagreements between the LSTM-CRF+WiNER(5M) system (columns) and the 2-stage approach (rows). The table also shows the percentage of times the latter system was correct. For instance, the bottom left cell indicates that, on 38 distinct occasions, the classifier changed the tag PER proposed by the native system to ORG and that is was right in 85% of these occasions. We exclude errors made by both systems, which explains the low counts observed (1.7% is the absolute difference between the two approaches).

We observe that in most cases the classifier makes the right decision when an entity tag is changed from PER to either LOC or ORG (86% and

	PER	LOC	ORG
PER	-	50% [12]	25% [12]
LOC	86% [20]	-	21% [28]
ORG	85% [38]	81% [19]	-

Table 7: Percentage of correctness of the 2-stage system (rows) when tagging a named-entity differently than the LSTM-CRF+WiNER(5M) (columns). Bracketed figures indicate the average number of differences over the out-domain test sets.

85% respectively). Most often, re-classified entities are ambiguous ones. Our approach chooses correctly mostly by examining the context of the mention. For instance, the entity *Olin* in example (a) of Figure 4 is commonly known as a last name. It was correctly re-classified as ORG thanks to its surrounding context. Replacing *its* by *his* in the sentence makes the classifier tag the entity as PER. Similarly, the entity *Piedmont* in example (b) was re-classified as ORG, although it is mostly used as the region name (even in Wikipedia), thanks to the context-based CONT and MIX features that identify the entity as ORG (0.61 and 0.63 respectively).

(a) ... would give [Olin]$_{\text{PER}\rightarrow\text{ORG}}$ access to its production processes ...

(b) Wall Street traders said [Piedmont]$_{\text{LOC}\rightarrow\text{ORG}}$ shares fell partly ...

(c) $\star$ ... performed as a tenor at New York City 's [Carnegie Hall]$_{\text{ORG}\rightarrow\text{LOC}}$.

Figure 4: Example of entities re-classified by our 2-stage approach.

Misclassification errors do occur, especially when the native system tagged an entity as ORG. In such cases, the classifier is often misled by a strong signal emerging from one family of features. For instance, in example (c) of Figure 4, both MIX — $p(\text{ORG}) = 0.39$ vs. $p(\text{LOC}) = 0.33$ — and EMB — $p(\text{ORG}) = 0.39$ vs. $p(\text{LOC}) = 0.38$ — features are suggesting that the entity should be tagged as LOC, but the CONT signal — $p(\text{LOC}) = 0.63$ vs. $p(\text{ORG}) = 0.1$ — strongly impacts the final decision. This was to be expected considering the simplicity of our classifier, and leaves room for further improvements.

8 Conclusion and Future Work

We revisited the task of using Wikipedia for generating annotated data suitable for training NER systems. We significantly extended the number of annotations of non anchored strings, thanks to coreference information and an analysis of the Wikipedia's link structure. We applied our approach to a dump of English Wikipedia from 2013, leading to WiNER, a corpus which surpasses other similar corpora, both in terms of quantity and of annotation quality. We evaluated the impact of our corpus on 4 reference NER systems with 6 different NER benchmarks. The LTSM-CRF system of (Lample et al., 2016) seems to be the one that benefits the most from WiNER overall. Still, shortage of memory or lengthy training times prevent us from measuring the full potential of our corpus. Thus, we proposed an entity-type classifier that exploits a set of features computed over an arbitrary large part of WiNER. Using this classifier for labelling the types of segments identified by a reference NER system yields a 2-stage process that further improves overall performance. WiNER and the classifier we trained are available at http://rali.iro.umontreal.ca/rali/en/winer-wikipedia-for-ner. As future work, we want to study the usefulness of WiNER on a fine-grained entity type task, possibly revisiting the simple classifier we resorted to in this work, and testing its benefits for other currently successful models.

Acknowledgments

This work has been partly funded by the TRIBE Natural Sciences and Engineering Research Council of Canada CREATE program and Nuance Foundation. We are grateful to the reviewers for their helpful comments.

References

Rami Al-Rfou, Vivek Kulkarni, Bryan Perozzi, and Steven Skiena. 2015. Polyglot-NER: Massive multilingual named entity recognition. In *Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, British Columbia, Canada*. SIAM.

Giuseppe Attardi. 2015. Deepnl: A Deep Learning NLP Pipeline. In *Proceedings of Workshop on Vector Space Modeling for NLP, NAACL-HLT*, pages 109–115.

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61–83.

Dominic Balasuriya, Nicky Ringland, Joel Nothman, Tara Murphy, and James R Curran. 2009. Named Entity Recognition in Wikipedia. In *Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources*, pages 10–18. Association for Computational Linguistics.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250.

Nancy Chinchor and Beth Sundheim. 2003. Message understanding conference (MUC) 6. *LDC2003T13*.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional LSTM-CNNs. In *Proceedings of the 54st Annual Meeting of the Association for Computational Linguistics*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics.

Abbas Ghaddar and Philippe Langlais. 2016a. Coreference in Wikipedia: Main Concept Resolution. In *CoNLL*, pages 229–238.

Abbas Ghaddar and Philippe Langlais. 2016b. WikiCoref: An English Coreference-annotated Corpus of Wikipedia Articles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Portorož, Slovenia.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable Modified Kneser-Ney Language Model Estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 690–696, Sofia, Bulgaria.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for Named Entity Recognition. *arXiv preprint arXiv:1603.01360*.

Joel Nothman. 2008. *Learning named entity recognition from Wikipedia*. Ph.D. thesis, The University of Sydney Australia 7.

Joel Nothman, James R Curran, and Tara Murphy. 2008. Transforming Wikipedia into named entity training data. In *Proceedings of the Australian Language Technology Workshop*, pages 124–132.

Kezban Dilek Onal and Pinar Karagoz. 2015. Named entity recognition from scratch on social media. In *ECML-PKDD, MUSE Workshop*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation. In *EMNLP*, volume 14, pages 1532–1543.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards Robust Linguistic Analysis using OntoNotes. In *CoNLL*, pages 143–152.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 1–40.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. Association for Computational Linguistics.

Alexander E. Richman and Patrick Schone. 2004. Mining Wiki Resources for Multilingual Named Entity Recognition In proceedings. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 1–9.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named Entity Recognition in Tweets: An Experimental Study. In *EMNLP*.

Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 142–147. Association for Computational Linguistics.

A. Toral and R. Munoz. 2006. A proposal to automatically build and maintain gazetteers for Named Entity Recognition by using Wikipedia. In *Proceedings of the EACL-2006 Workshop on New Text: Wikis and blogs and other dynamic text sourcesEACL Workhop on NEW TEXT-Wikis and blogs and ther dynamic text sources*, pages 56–61.