Structured Named Entity Recognition by Cascading CRFs

Yoann Dupont[1,2], Marco Dinarelli[1], Isabelle Tellier[1], and Christian Lautier[2]

[1] Laboratoire Lattice, UMR 8094 CNRS, 1 rue Maurice Arnoux, 92120 Montrouge
[2] Expert System France, 207 rue de Bercy, 75012 Paris

Abstract. NER is an important task in NLP, often used as a basis for further treatments. A new challenge has emerged in the last few years: *structured* named entity recognition, where not only named entities must be identified but also their *hierarchical components*. In this article, we describe a cascading CRFs approach to address this challenge. It reaches the state of the art while remaining very simple on a structured NER challenge. We then offer an error analysis of our system based on a detailed, yet simple, error classification.

Keywords: machine learning, structured named entity recognition, CRF, Quaero

1 Introduction

In this paper, we present a linear CRF cascade approach for structured named entity recognition (SNER) on Quaero v1 and v2 corpora, used in the ETAPE evaluation campaigns [10]. Named Entity Recognition (NER) is a fundamental NLP task, its structured variant being increasingly popular. We can overall distinguish two main approaches used to address this task, the first one being cascading multiple annotations with either the same or different methods. In this respect, we can cite [19], which cascaded rules in order to gradually build the structure. We can also cite [5], where a CRF and a PCFG were used, the former giving the leaves while the latter built the rest of the tree. And finally [22], the winner of ETAPE, used one CRF per entity type, for a total of 68 CRFs, and then aligned their annotations. The second approach to annotate tree-structured named entities is to directly retrieve the structure, as was done by [20], who used partial annotation rules for predicting beginnings and ends of entities and then built the tree in one pass. Finally, we can cite [8], who used a tree-CRF to learn nested biomedical entities on the GENIA corpus [14].

Cascading linear CRFs have also been applied for syntactic parsing, as did [25]. At each step, they retrieved chunks and then only kept their respective heads for the next iteration until only one chunk covering the whole sentence was found (with the class "sentence"). The tree was then reconstructed by simply unfolding chunks at each step. In this paper, we design a new, more general and effective cascade of CRFs adapted to the ETAPE evaluation campaign (sections 2 and 3), evaluate its efficiency and analyse its errors (section 4) and finally conclude (section 5).

2 Structured Named Entity Recognition

2.1 Named Entity Recognition

NER is a very important NLP task, often used as the starting point of many others, such as relation extraction [2], entity linking and coreference resolution [4,7,12].

Since their definition in the MUC-6 [11], named entities have been integrated into more and more refined classifications, covering more elements of different nature and/or refining the grain of already defined typologies [6,24]. The need for structuration in named entities appeared early. The first available corpora where this need was taken into account came out with an imbrication structure where the same entity set was used along different annotation layers, applied to longer and longer sequences. It is for instance the case for the SemEval'2007 [18] task 9 corpora. To our knowledge, one of the first corpus providing real structured named entities is Quaero [23], which we will use for our experiments.

2.2 Quaero Corpus

The Quaero corpus is made of French transcribed oral broadcast news. Two annotation variants (v1 and v2) have been applied to the same data. Their main characteristics are given in the table 1, from which we can see that there are 60% more annotations in Quaero v2 compared to Quaero v1 (v1 annotations thus probably keep silent on many entities). The specificity of the Quaero typology is that it integrates two kinds of annotations: *types* (that we will call *entities* for sake of clarity) and *components*. *Entities* follow the common named entity definition: they can be a location, a person, an organisation, an amount, etc. The different Quaero *entities* are shown in figure 1. *Components*, as their name suggests, are parts of an entity. For example, a person has a first and/or last name, an absolute date may have a year and/or a day and/or a month. This means that a *component* cannot be at the top level of the hierarchy. There are 27 of them, 10 of which are transversal, meaning that they can be components of different entity types.

The main Quaero difficulties lie in its wide coverage named entity definition, Quaero considering a lot of common nouns as named entities, its tree structure named entity and the fact that it is oral transcription.

Some differences between the typologies of Quaero v1 and v2 are shown in Figure 2. Amongst the most notable differences between the two versions, there is the disappearance of organisation sub-types, namely *org.ent* (companies), *org.adm* (organisations) and *org.other* (other organisations), replaced by *org.ind* (individual organisation) and *org.coll* (a collection of organisations). Many *kind* components were refined: functions, for example, are components on their own in v2. Some changes go along those previously cited: in v1, *function* and *person* were two different entities, in v2 they are one. This echoes the change of some *kind* components to *function*, a function being a component of a person in v2, whose spans were enlarged accordingly.

		training	test
documents		188	18
tokens		1,291,225	108,010
components	v1	146,405	8,902
	v2	255,498	13,612
entities	v1	113,885	5,523
	v2	161,984	8,399

Table 1. Statistics on the Quaero Train and Test Sets

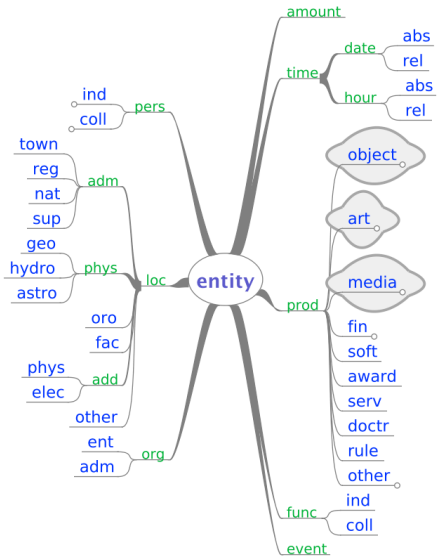

Fig. 1. Quaero v1 *Entities*

Quaero offers a very large number of annotations of very different natures, many entities being noun phrases without a proper name. It is for example the case for amounts, like in *deux incendies*[4] or *des historiens* [5], but not in sport results or administrative language (e.g. *affirmation 22*[6] in Quaero guidelines). The generic nature of some entities make them sometimes hard to grasp.

While most Quaero entities have a depth of 2, there is no limit in Quaero's definitions of how deep an entity can be: we found that the deepest Quaero entity was of depth 9 (we cannot show it here for space issues). We also checked for overlapping entities having the same type, which is an argument for using cascading annotations. We found about 300 examples in the training set, a little more than 1 per thousand annotations. The system used by the winner of the

[3] The President of Burkina Faso Blaise Compaore
[4] French for *two wildfires*
[5] French for *some historians*
[6] French for *assertion 22*, Quaero annotation guidelines being written in French

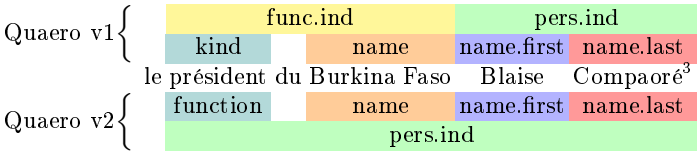

Fig. 2. Some Differences between Quaero v1 et v2

challenge [22], who used binarized CRFs (one per type), is unable to model this kind of structures. Given that most components are entity-specific and given the two phenomena previously mentioned, cascaded annotation approaches are an effective way to deal with Quaero specificities, while allowing to recognize such embedded structures. In this work we obtain such an effective modeling using a cascade of linear-chain CRFs.

3 Linear-chain CRF cascade

Linear-chain CRFs [15] are discriminative probabilistic graphical models modeling sequential dependencies. One of the most effective implementations of linear-chain CRFs is Wapiti [16], which was used for our experiments.

The principle of a linear-chain CRF (or of any other linear-chain model) cascade for structured annotation is very simple, yet has proven to be effective for syntactic parsing [21,25]. A basic overview is that one or multiple chunking models are used repeatedly until no more additional information is found. Taking syntactic parsing as an example, it means that, at each step, one chunk if found (NP, VP, etc.), until there is only one chunk called "S" (for sentence) left, that spans over the entire sequence. Our contribution is an adaptation of this CRF parsing technique for structured named entities to better fit the particularities of the task at hand. The main problem a classic parsing algorithm has to deal with when applied to named entities is the overabundance of "out" labels (words that are not part of an entity). Using previous algorithms as they are, a lot of passes would consist in parsing "out" labels, which would be suboptimal. We adapted the algorithm as follows: since we do not want to fully parse the sentence, we will stop as soon as no new entities are found at a given step. For the Quaero corpus, the simplest instance of cascade would consist in training two CRFs, one for *components* and the other one for *types*, used alternatively to annotate entities layer by layer. An example of a layered annotation is illustrated in Figure 2. This approach can be generalized to any number of CRFs. This approach was not proposed by any ETAPE constestant. The closest systems would be: 1. one constestant used fixed-depth CRF for *entities* and retrieved *components* using a rule-based approach and 2. the system of ETAPE's winner, who used binarized CRFs (one per type). Neither system use any kind of recursion, making ours more general and closer to Quaero's entity structure.

Our method is "cumulative", being in this respect somewhat comparable to the ones described in [21,25]. At each step, when an entity of length two or more

is found, it is merged into a single token. Previous methods, typically used for syntactic parsing, substitute the sequence of tokens by the head of the chunk, so they only keep the most relevant token. For named entity chunking, the concept of head word does not seem so natural and useful information could easily be lost. For an overview of heuristics used for cumulating tokens, see section 4.1.

Quaero entities may be very deep, we then need some recursion in our annotation scheme. The simplest way to achieve this is to have one model that would annotate components and one that would annotate entities. However, entities may be components of other entities. To model full annotations, we use two main passes: the first one being a "no context" annotation, where a first annotation has to be made with no additional information. The second one is a "context aware" annotation, where a context can be seen by the current CRF. Quaero entities also have the property that a component will always have a type as an ancestor in a parse tree. To model this property, we divide each pass into two annotations, each one being done with a specific CRF. This gives us a total of four CRFs that will be launched, following the the algorithm 1 (for sake of simplicity, we left out the entity aggregation to one token and rebuilding of base text). The first two CRFs (leaf) are called once to give a starting context to the other two (upper), which will be successively called until there is no more additional annotation. For specific features used in our models, see section 4.1. We have observed in our experiments that using this approach we were able to manage annotations up to a depth of 6. Our approach is thus able to model recursion, improving the more naive fixed-depth CRF used during the ETAPE evaluation campaign.

To illustrate with the example of figure 2, CRF1 and CRF2 would annotate the first two levels as illustrated. CRF3 and CRF4 would not find any *component* or *entity* above, and would then stop.

Algorithm 1 The base algorithm for CRF cascade

function CRFCascade($Corpus, leafC, leafE, upperC, upperE$)
▷ *C are models for components. *E are models for entities.
annotations ← ∅;
currentAnnotations ← ∅;
annotations ← annotations ∪ annotate(Corpus, leafC);
annotations ← annotations ∪ annotate(Corpus, leafE) ;
newAnnotations ← (annotations ≠ ∅);
while newAnnotations **do**
annotations ← annotations ∪ currentAnnotations;
currentAnnotations ← ∅;
currentAnnotations ← currentAnnotations ∪ annotate(Corpus, upperC);
currentAnnotations ← currentAnnotations ∪ annotate(Corpus, upperE);
newAnnotations ← (currentAnnotations ∩ annotations ≠ ∅);
end while;
return annotations;
end function;

4 Results

In this section, we present the results reached with our method. We will first compare the results we obtained on Quaero v1 with those of the contestants of the ETAPE evaluation campaign, as a first evaluation of our method. We will then analyse the errors it made on Quaero v2, for which no other result has been published yet.

4.1 Features and Performances

Every feature detailed here is applied on a window of two words before to two words after. We considered different sets of features to evaluate the importance that some have compared to others.

For our baseline, we only used word-specific features: not a single lexicon is used, no tagging or lemmatisation is performed. The features used are the shapes of the words, their prefixes and suffixes up to a length 5 and a variety of boolean features such as “does the word start with an uppercase?” or “is the word a number?”. This baseline will then be enriched with other (more or less specific) features, to measure their impact.

We first added the outputs of basic syntactic analyses, namely lemmatisation, PoS and chunking. This model is called "+syntax". As can be seen, adding this information leads to an important quality loss. This is probably due to the fact that they do not provide any new information (lemmas) or are not precise enough (PoS, chunking).

It is commonly known that the verb is the most important syntactic unit of a sentence. Verbs could be used to disambiguate between various entities and help improving recall on unknown entities, as the same verbs could be used for entities of the same type. We added, for each word, the previous and next verb found in the sentence. French uses auxiliaries in some tenses, which precede the verb: in this case, we took the first non-auxiliary verb. This provides the "+verb" model.

We then used a full set of features, containing all the previous features described above. We also added "word classes": these classes are obtained by substituting uppercase letters by “A”, lowercase letters by “a”, numbers by “0” and everything else by “x”. The “brief” alternative version of this feature consists in applying the same substitutions, but on contiguous sequences of characters of the same class. For example, the first name “Jean-Pierre” would become “Aaaax-Aaaaaa” as a word classe and “AaxAa” as brief word class. This allows to represent words in a condensed fashion that is far more general than lemmas. We also have some basic chunk-based patterns (sequences of prepositional phrases following some keyword) which simulate “rules-based” entity recognizers. We used some gazetteers extracted either from Wikipedia or from internal tools, mainly first names, last names, locations and companies. Quaero being an oral corpus, we also removed discourse markers using the list defined by [3], but only the non

ambiguous ones such as "euh"[7] or "enfin bref"[8]. We did not remove, for example, "ben" [9] as it could also be a part of an Arab name. We removed repeated words with the exception of "nous nous"[10] or consecutive numbers. We considered those markers as part of an entity if they were in the middle of it, but not otherwise.

When doing accumulation, a lot of interesting information may be lost. To limit this loss, we defined some heuristic rules based on which information the feature is supposed to extract. Examples of such rules are given in table 2

We also tried a top-down approach: finding entities first, then components. While it is relatively easy to retrieve entities when their components have been identified, components themselves may be difficult to identify: some components, such as *kind*, *name*, *extractor*, *range-mark*, *object*, tend to be ambiguous as they can either cover entities of very different natures, or be very contextual and appear in conjunction with others (an *extractor* is never isolated, for example). Their identification could be eased by first retrieving the entities that cover them, giving more useful context to the CRF.

feature	example
word	12 January → 12_January
character classes	00 Aaaaaaa → 00_Aaaaaaa
first is upper?	12 January → false
has number?	12 January → true
is number?	12 January → false

Table 2. Examples of Heuristic Rules of Accumulation

The metric used to measure performances in ETAPE evaluation campaign is a modified version of the Slot Error Rate (SER) [17], which is the ratio between errors made by the system and the number of slots in the reference (N). The errors in the original SER are divided into three categories: substitutions (S), deletions (D) and insertions (I). Deletions measure the silence of a system (slots in the reference which cannot be aligned to suggestions of the system), while insertions measure its noise (slots in the system's suggestions which cannot be aligned to a reference slot), substitutions are the rest of precision errors. ETAPE used a weighted SER: pure type errors (S_t) and pure boundary errors (S_b) were counted as half an error, while type and boundary errors (S_{t+b}) were counted as a full error, which gives the equation 1. It is the measure we used.

$$SER = \frac{D + I + S_{t+b} + 0.5 * (S_t + S_b)}{N} \quad (1)$$

The results reached with our cascade of CRFs with different sets of features are compared with those of the top 5 contestants of the ETAPE campaign in the

[7] French for "err"
[8] French for "anyway"
[9] which can stand for "well" in French oral discourses
[10] which can be a correct sequence in French

tables 3 (SER being an error rate, the lower the best). Had we participated in ETAPE campaign, our model would have reached second position with our baseline CRF cascade, which does not include any kind of morphosyntactic analysis, dictionary or any other external resource. Top competitors in the ETAPE campaign used some external tools. [5] used WMatch [1,9], ETAPE's winner [22] used dictionaries along mined trigger words (words that have high mutual information with output classes) and a number discretiser. Our approach is competitive, as our baseline would have ranked second without using any such resource. We also have a significant quality improvement using our cascade compared to using only a naive *two levels* CRF cascade. We did not manage to improve our baseline on Quaero v1, going from slightly worse to significantly worse, the worst being when the full set of features was used. That last experiment had roughly twice the noise of the baseline. As seen in section 2.2, this noise may actually be corrections of the silence due to incomplete annotations.

Contestant	method	SER
3	CRF	**33.8**
8	CRF+PCFG	36.4
10	rules	42.9
5	CRF	43.6
4	rules	55.6

Our results	SER
baseline	**35.5**
+syntax	37.0
+verbs	37.4
full set	43.3
two levels	37.0
top-down	37.1

Table 3. On the left, results of ETAPE contestants. On the right, our results.

experience	SER	micro F1	macro F1
our baseline	**33.2**	**73.1**	**54.2**
+verb	33.7	72.9	51.4
full set	34.8	72.3	53.2

Table 4. Our Best Results on Quaero v2.

As can be seen in table 4, we obtain better results on Quaero v2 than on Quaero v1, due to improved typology and a more thorough human annotation. We also see that adding neighboring verbs has a detrimental effect on the quality of the annotation, no matter the experiment. Dictionaries, surprisingly, also had a detrimental effect on our results, but far smaller on Quaero v2 than on Quaero v1, which shows that a lot of the noise induced by dictionaries in Quaero v1 were actually entities missed by the annotators (SER penalizes more systems that are noisy). Looking at macro F1-scores in Table 4, we can see that the full set of features yields better results, and that the lower micro F1-score is due to the imbalance in data set, as told in section 2.2. Table 5 shows some examples of difference in terms of F1-scores between the baseline and the other experiments,

displaying why using the full set leads to worse results: the quality on *amount*, which is disproportionately represented, decreased significantly.

	+verbs	full set
town	-0.6	+7.1
org.ind	-1.4	+1
amount	+0.8	-2.2

Table 5. Some Entity-specific F1-score Differences Compared to the Baseline

Despite these results, it is obvious that our system can still be improved. Since SER as a unique measure is not very informative, we make a more detailed analysis in the next section, trying to find some hints on where we can get improvements. Since there are some papers on Quaero v1, but none to our knowledge on Quaero v2, we will focus our error analysis on the latter.

4.2 Error analysis

SER, as well as micro F1-score, is a measure that tends to favor most frequent entities as they carry more weight on the global metric than less frequent ones. Displaying scores by entity may allow to know on which ones the system performs better, but does not give an accurate view on where best gains can be made. To make up for this, we suggest a quantification of the shortfalls of our system in Table 6. These shortfalls are the number of F1-score points the system would gain if it were 100% accurate on a specific entity.

We see in the tables that shortfalls gather on leaves or on major types (*amount*, *org*, *pers*). If we perfectly annotated the entities of these tables, we would obtain about 90 in absolute F1-score. We can see that gains are hard to make by focusing on a single entity: if we wanted to gain a single F1-score point that way, this would equate to gaining 10 points on *name* components, 20 on *org.ind* or 24 or *kind*. However, just as errors propagate, corrections would also propagate: *name* being a component of multiple entities, corrections on that component would also spread on the entities above it and would improve scores on multiple entities (for example *org*, *loc* and *pers* where most ambiguities happen at the component level).

To ease error analysis, we capitalized on the Quaero refinements on SER, giving us 5 kinds of errors: type errors, boundary errors, type+boundary errors, noise and silence.

As illustrated on the table of table 7, the main problem of our system is its silence, amounting to more than 50% of the system's errors, 19% of reference annotations not having a suggestion made by the CRF.

Now, we detail the most common errors made by our system. First, examples of such errors are given in the Table 8 for *components*. Errors on entities being mainly propagated, we will focus on component errors.

entity	F1-score	F1 gain if perfect
name	81.48	2.28
org.ind	65.12	2.25
amount	75.48	2.17
kind	51.20	1.97
qualifier	49.51	1.73
object	76.03	1.7
pers.coll	59.91	1.68
pers.ind	78.05	1.4

Table 6. Entities with the Highest Shortfalls on Global F1-scores

error kind	proportion (%)
type	8.0
boundary	11.7
type+boundary	6.2
noise	21.6
silence	52.5

Table 7. Raw Percentage of the Various Errors

error	description
boundary	– unfrequent variation of frequent entity – adjective or prepositional phrase
type	*kind* vs *func* (+human errors?) *name* / *kind* : some *name* become *kind* with other components (in gold)
noise	*val*: wrong PoS on "des", "de" et "d' "[11] (+human errors?) *object*: common nouns and sports results *name*: known components → country, val (numbers), relative time
silence	– *val* : not numbered amounts – *qualifier* : missed if qualified component is missed – *name* : forgotten on relative times – *kind* : polysemic common nouns

Table 8. Error overview on components

As illustrated on the chart of Figure 3, most type errors involve either *func*, *kind* or *name*. Going from Quaero v1 to Quaero v2, some *kind* components have been replaced by *func* (cf. Figure 2): they are closely related but there are also some possible human errors which could explain in part the confusion between the two. Some errors come from *name* morphing into *kind* in presence of other components (ex: a country's government). CRFs seem to have trouble modeling this "isolation" phenomenon. Still on the government example, there is a volume

[11] "de" and "des" in French may be partitive, possessive (not annotated) or complement (annotated).

disparity between gold annotations and what the CRF yields: while it is mostly annotated *name*, this annotation only amounts to 20% of the CRF's output. Maybe some post-processing rules could help correct this kind of errors.

Silence errors are mainly made on Quaero components, amounting to nearly 60% of all silence errors. They are mainly made on *val*, *object*, *kind* and *qualifier*. *Object* being an *amount* component, it is accompanied with a *val*, most likely those errors are linked to each other, even though we did not manage to quantify the phenomenon. Errors on *qualifier* are nearly always contextual, as it never appears alone. Most silent qualifiers are so because the component they qualify was not identified either. This allows to think that the CRF managed to "understand" this structural constraint, meaning that those silences will most likely be corrected if we manage to catch the component they qualify.

Boundary errors on components are usually of length 1 or 2 and seem equally distributed between additions and deletions, they mostly are adjectives or prepositional phrases. When it comes to entities, boundary errors tend to be larger on overall, this is due to the propagation of two kinds of errors: first the boundary errors on some components will cause an entity to have a boundary error also. Second, a silence error on a component can lead to a boundary error in an entity, for example when a first/last name is not identified at a component level, but the person is still identified.

Most noise errors are on components such as *val*, *object*, *kind*, *qualifier* and *name*. Nearly 80% of those noise errors are on components whose form was observed on the training corpus. While some are most likely human errors, such as countries and proper names, some others are more contextual and may indicate an overfitting of the CRF, that just took those components "at face value".

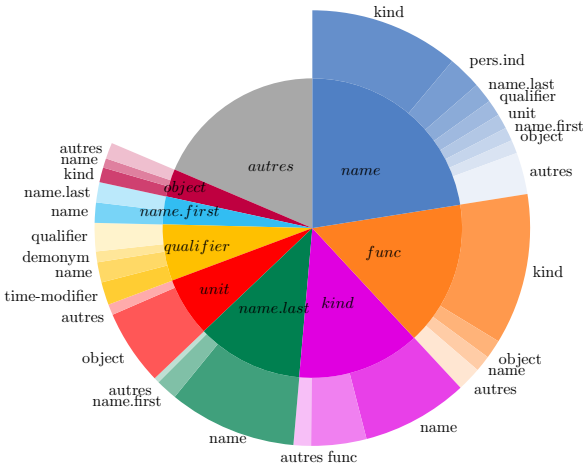

Fig. 3. Type Errors on Components

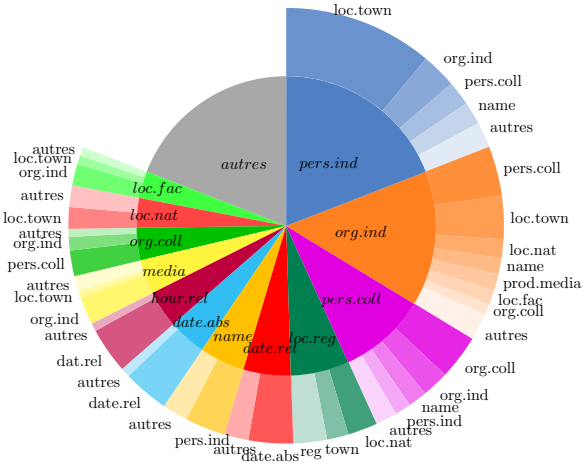

Fig. 4. Type Errors on Entities

As previously showed, most errors on entities seem to originate from errors made on previous levels. To check this assertion, we tried a run using the reference components instead of using a CRF to annotate leaf level components in algorithm 1: the SER dropped to 6.3%, a result coherent with the one stated by [5], who made the same test (Table 4). We plan to isolate non-propagated type-specific errors to analyse them specifically in further research. This last test provides a strong proof that we should focus more on components of the first level, especially for common nouns that tend to be more ambiguous than proper nouns. We also need to model some "horizontal" structuration better: some components work "symbiotically" with others, such as *val* and *object*. Type errors also showed the need for a better disambiguation between the different *name* components.

5 Conclusion

In this article, we have described a general method for structured named entity recognition using a cascade of linear-chain CRFs. We have given a generic procedure that we adapted to best fit the architecture of Quaero named entities. We showed why this specific architecture was justified; it gave promising results, while remaining simple. While we did not manage to improve the current state-of-the-art on Quaero v1, we nonetheless showed that our approach has competitive performances.

We tried to characterize the most common errors and quantified the different shortfalls of our system, which gave us some insights on how to improve it and even found potential human errors. This process sadly lacks in automation. We could compute an estimation of the propagated errors on types by checking if a component below it has the same error type. We could also check the merits

of our approach by comparing it to a single CRF that would learn only the top-level entities: by comparing the two, we could see errors made by one and not the other, or by both of them. We could also compare the SER score with a recent metric named "Entity Tree Error Rate" (ETER) [13], a metric based on the SER but aims to better take into account structuration.

We plan to continue our research, especially by integrating more efficient models, by focusing on annotating common nouns and how to model context, which we think are the two most important tasks if we want to improve results on Quaero. We also plan to use the hierarchy of the Quaero entity types to our advantage: we could first learn a coarse grain CRF (ex: *pers* instead of *pers.ind* and *pers.coll*) which would be followed by a fine grain CRF that would assign the various subclasses. This could improve the disambiguation between the different subtypes of *name*.

References

1. Guillaume Bernard, Sophie Rosset, Olivier Galibert, Gilles Adda, and Eric Bilinski: The limsi participation in the qast 2009 track: experimenting on answer scoring. In Workshop of the Cross-Language Evaluation Forum for European Languages, pages 289–296. Springer (2009)
2. Razvan C Bunescu and Raymond J Mooney: A shortest path dependency kernel for relation extraction. In Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, pages 724–731. Association for Computational Linguistics (2005).
3. Catherine Chanet: Fréquence des marqueurs discursifs en français parlé: quelques problèmes de méthodologie. Recherches sur le français parlé, 18:83–106.
4. Pascal Denis, Jason Baldridge, et al.: Global joint models for coreference resolution and named entity classification. Procesamiento del Lenguaje Natural, 42(1):87–96 (2009)
5. Marco Dinarelli and Sophie Rosset: Models cascade for tree-structured named entity detection. In Proceedings of 5th International Joint Conference on Natural Language Processing, pages 1269–1278, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing (2011)
6. George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie Strassel, and Ralph M Weischedel: The automatic content extraction (ace) program-tasks, data, and evaluation. In LREC, volume 2, page 1 (2004)
7. Greg Durrett and Dan Klein: A joint model for entity analysis: Coreference, typing, and linking. Transactions of the Association for Computational Linguistics, 2:477–490 (2014)
8. Jenny Rose Finkel and Christopher D Manning: Nested named entity recognition. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1, pages 141–150. Association for Computational Linguistics (2009)
9. Galibert, Olivier: Approches et méthodologies pour la réponse automatique à des questions adaptées à un cadre interactif en domaine ouvert. Diss. Université Paris Sud-Paris XI, 2009.

10. Guillaume Gravier, Gilles Adda, Niklas Paulson, Matthieu Carré, Aude Giraudel, and Olivier Galibert: The etape corpus for the evaluation of speech-based tv content processing in the french language. In LREC-Eighth international conference on Language Resources and Evaluation (2012)
11. Ralph Grishman and Beth Sundheim: Message understanding conference-6: A brief history. In COLING, volume 96, pages 466–471 (1996)
12. Hannaneh Hajishirzi, Leila Zilles, Daniel S Weld, and Luke S Zettlemoyer: Joint coreference resolution and named-entity linking with multi-pass sieves. In EMNLP, pages 289–299 (2013)
13. Mohamed Ben Jannet, Martine Adda-Decker, Olivier Galibert, Juliette Kahn, and Sophie Rosset Eter: a new metric for the evaluation of hierarchical named entity recognition. In Ninth International Conference on Language Resources and Evaluation, pages 3987–3994 (2014)
14. J-D Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii: Genia corpus—a semantically annotated corpus for bio-textmining. Bioinformatics, 19(suppl 1):i180–i182 (2003)
15. J. Lafferty, A. McCallum, and F. Pereira: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proceedings of ICML 2001, pages 282–289 (2001)
16. Thomas Lavergne, Olivier Cappé, and François Yvon: Practical very large scale CRFs. In Proceedings of ACL'2010, pages 504–513. Association for Computational Linguistics, July (2010)
17. John Makhoul, Francis Kubala, Richard Schwartz, Ralph Weischedel, et al.: Performance measures for information extraction. In Proceedings of DARPA broadcast news workshop, pages 249–252 (1999)
18. Lluís Màrquez, Lluis Villarejo, M. A. Martí, and Mariona Taulé: Semeval-2007 task 09: Multilevel semantic annotation of catalan and spanish. In Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007), pages 42–47, Prague, Czech Republic, June. Association for Computational Linguistics (2007)
19. Denis Maurel, Nathalie Friburger, Jean-Yves Antoine, Iris Eshkol, and Damien Nouvel: Cascades de transducteurs autour de la reconnaissance des entités nommées. Traitement automatique des langues, 52(1):69–96 (2011)
20. Damien Nouvel, Jean-Yves Antoine, Nathalie Freeburger, and Arnaud Soulet: Fouille de règles d'annotation partielles pour la reconnaissance des entités nommées. In In Procs of TALN (2013)
21. Adwait Ratnaparkhi: A linear observed time statistical parser based on maximum entropy models. arXiv preprint cmp-lg/9706014 (1997)
22. Christian Raymond: Robust tree-structured named entities recognition from speech. In 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pages 8475–8479. IEEE (2013).
23. Sophie Rosset, Cyril Grouin, and Pierre Zweigenbaum: Entités nommées structurées: guide d'annotation Quaero. LIMSI-Centre national de la recherche scientifique (2011)
24. Satoshi Sekine and Chikashi Nobata: Definition, dictionaries and tagger for extended named entity hierarchy. In LREC (2004)
25. Yoshimasa Tsuruoka and Sophia Ananiadou: Fast full parsing by linearchain conditional random fields. In In Proc. of EACL (2009)