Named Entity Recognition in the Medical Domain with Constrained CRF Models

Charles Jochim
IBM Research - Ireland
Dublin, Ireland
charlesj@ie.ibm.com

Léa A. Deleris
IBM Research - Ireland
Dublin, Ireland
lea.deleris@ie.ibm.com

Abstract

This paper investigates how to improve performance on information extraction tasks by constraining and sequencing CRF-based approaches. We consider two different relation extraction tasks, both from the medical literature: dependence relations and probability statements. We explore whether adding constraints can lead to an improvement over standard CRF decoding. Results on our relation extraction tasks are promising, showing significant increases in performance from both (i) adding constraints to post-process the output of a baseline CRF, which captures "domain knowledge", and (ii) further allowing flexibility in the application of those constraints by leveraging a binary classifier as a pre-processing step.

1 Introduction

With the number of articles indexed by MEDLINE/PubMed exceeding one million articles per year[1], manual consumption of published medical literature is no longer practical and researchers are increasingly turning to automated techniques to quickly identify and process relevant medical knowledge (e.g., literature-based discovery (Hristovski et al., 2006)). Our overall project's objective is to semi-automate the construction of decision support models by generating probabilistic graphical models of medical conditions and their associated risks based on the academic literature (Deleris et al., 2013). An essential task in this project consists of extracting from medical papers any relations mentioning (i) dependence or independence between variables and (ii) probability statements indicating the strength of a relationship. For both types of relations, we have approached the entity extraction step as a sequence labeling problem. We then rely on a set of rules to construct relations from the entities extracted. This paper focuses specifically on the entity extraction step which we describe in more detail in the following section. We then proceed in Section 3 with details about our suggested constraint enforcement procedures. Section 4 reports details about the experiments, with numerical results reported and discussed in Section 5.

2 Dependence Relation and Probability Statement Extraction

2.1 Dependence Relation Extraction

The first task concerns identifying dependence relations between pairs of potential variables mentioned in text. Dependence and independence here are to be understood as defined by probability theory where A depends on B iff $\Pr(A) \neq \Pr(A|B)$. As independence statements seldom occur in the literature, we focus predominantly on dependence in this paper and define independence as the negation of dependence. Our choice to use the term "dependence" to describe the task may lead to some ambiguity in the NLP world yet from a probability perspective, it is a precise characterization. Thus we caution the reader that our use of the word "dependence" is exclusively related to its meaning in probability theory. We structure dependence relations as being composed of two variables (variable A and variable B) which are interchangeable, one influence term I and an optional negation.[2] As an example, from the sentence "For endometrial cancer, body mass index represents a major modifiable risk factor; about half of all cases in

[1] https://mbr.nlm.nih.gov/Background.shtml

[2] Negation enables us to capture independence though we ignore that last element for the remainder of the paper.

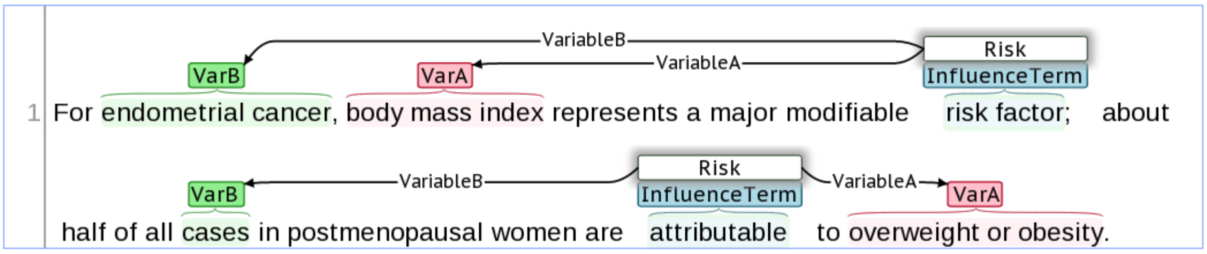
VariableB
VariableA
VarB
VarA
Risk
InfluenceTerm
1
For endometrial cancer, body mass index represents a major modifiable risk factor; about
VariableB
Risk
InfluenceTerm
VariableA
VarB
VarA
half of all cases in postmenopausal women are attributable to overweight or obesity.

Figure 1: Dependence relation example in brat (Stenetorp et al., 2012)

postmenopausal women are attributable to overweight or obesity," we extract two structured relations. The first involves variable A, *body mass index*, variable B, *endometrial cancer*, and an influence term, *risk factor*, and the second has a variable A, *overweight or obesity*, variable B, *cases* and influence term, *attributable* (see Figure 1).

2.2 Probability Statement Extraction

The second task focuses on extracting probability statements composed of probability terms (numbers) along with variables A (the conditioned variable) and B (the conditioning variable). Together they form a conditional probability statement, i.e., $\Pr(A|B) = x$. For instance, from the sentence "Malaysian women have a one in 20 chance of developing breast cancer in their lifetime," we want to extract the probability number, *one in 20*; the variable A, *developing breast cancer in their lifetime*; and the variable B, *Malaysian women*, which could be represented as $\Pr(\textit{developing breast cancer in their lifetime}|\textit{Malaysian women}) = 0.05$.

2.3 Insights into the Extraction Task

Probability terms, and to a lesser extent influence terms, are fairly regular and therefore more easily classified, while risk variables A and B are heterogeneous and exhibit a broad semantic variety. "1977-1990", "breast cancer", "homozygous carriers", "> or = 10 years", and "younger than age 35" are some examples of different variables taken from our corpus.

Risk variable identification presents multiple challenges. Consider the example "Carriers of the AC haplotype, which represents the variant alleles of both SNPs, were at an increased risk (OR = 1.41, 95% CI 1.09-1.82)." We have an odds ratio probability term "OR = 1.41" and two variables. However, determining the boundaries of the variables is not straightforward. Should the conditioning variable be the whole subject including the relative clause, only "Carriers of the AC haplotype," or even simply "AC haplotype"? We sidestep this issue in our current work because these variables will later be clustered and aggregated into variable groups, e.g., a variable group "breast cancer" could include mentions of "breast cancer", "ER+", "breast carcinoma" and others.

Finally, the distinction between variables A and B is essential when constructing a probabilistic statement (while not significant for dependence extraction) but it can be problematic to distinguish the two when extracting them from text. Overall, the variable identification task has proved quite challenging. In fact, our interest in exploring constraints is motivated by preliminary results which hinted, as we will explain in more details in the next section, that, provided a probability number or an influence term is detected, we should further encourage the algorithm to search for the other associated variables.

3 Methodology

We approach the entity extraction (probability term and variables on one side, influence terms and variables on the other side) as a sequence labeling problem. We want to identify the best sequence of labels y for a sequence of tokens x comprising a sentence. The labels in our vocabulary are O ("outside"), A (variable A), B (variable B) and, depending on the specific task, P (probability term) or I (influence term). We initially propose using conditional random fields (CRF) (Lafferty et al., 2001) for this task as they have been successful for other related NLP sequence labeling tasks (Sha and Pereira, 2003; Settles, 2004). We start with a linear-chain CRF:

$$\Pr(y|x) = \frac{1}{Z_x}\exp(\sum_{t=1}^{T}\sum_{k}\lambda_k f_k(y_{t-1}, y_t, x_t))$$

Algorithm 1 Dependence Extraction Constraints

```
if influence term $> 0$ then
    ensure at least one $A$ and one $B$ label
else if $A > 0$ then
    ensure at least one $B$ label
else if $B > 0$ then
    ensure at least one $A$ label
end if
```

Algorithm 2 Probability Extraction Constraints

```
if probability term $= 0$ then
    ensure no labeled entities
else
    ensure at least one $A$ and one $B$ label
end if
```

where T is the number of observations indexed by t, k indexes the feature function f_k and weight λ_k, and $Z_{\mathbf{x}}$ normalizes over the entire input sequence, $Z_x = \sum_y \exp(\sum_{t=1}^{T} \sum_k \theta_k f_k(y_{t-1}, y_t, x_t))$.

The CRF performs satisfactorily leading to higher precision classification of the labels than recall. However, it is not able to capture some information that we know to be true. For instance, for these corpora, if there is a labeled influence or probability term, 92% and 99% of the time, respectively, there are co-occurring variables in the same sentence. Thus for probability statement extraction, if we have a sentence with a detected probability term, which can be reliably classified (with F_1 scores around 74), then we want to enforce the presence of both variables A and B. B is theoretically optional, as it is possible to find marginal probability, i.e., statements with an empty conditioning set. In practice, given our domain, we choose to impose the presence of at least one B label for each sentence with a P label. By contrast, for a sentence without a probability term, we want to discard such output altogether. In the case of dependence statement extraction, the situation is similar, if we detect an influence term or a variable, we want to force the system to find the other relevant pieces of the relations. The constraints we apply to each of our two tasks are summarized in structured language in Algorithm 1 and Algorithm 2.

While it is straightforward to discard sentences that do not contain a given label, constraining the output with statements to enforce the presence of at least one type of label given the presence of another label, is less obvious. We address it by complementing the initial classification with further steps to enforce the constraints. We have previously explored different approaches, varying in complexity, for finding the most likely sequence while applying these constraints (Deleris and Jochim, 2016). Here we settle on the constrained approach inspired by Culotta and McCallum (2004) that uses posterior conditional probability in the CRF decoding.

3.1 Notation

We denote by y a vector of labels associated with observations x (tokens). Let T be the number of observations $(x_1, ..., x_T)$ then y also contains T elements $(y_1, ..., y_T)$ which we index by t and $y_t \in \{A, B, P, I, O\}$ $\forall t = 1 : T$.

Let $y^* = \text{argmax}_y \Pr(y|x)$ denote the output of applying Viterbi decoding to our observations x. To describe the proposed extensions, we then introduce the following variables : $t_P = \{t \in [1 : T] : y^*_t = P\}$, the indexes in the initial output corresponding to the label P. $t_I = \{t \in [1 : T] : y^*_t = I\}$, the indexes in the initial output corresponding to the label I. $t_A = \{t \in [1 : T] : y^*_t = A\}$, indexes corresponding to the label A. $t_B = \{t \in [1 : T] : y^*_t = B\}$, indexes corresponding to the label B. Finally we denote by t_O the set of unspecified indexes, i.e., $t_O = \{t \in [1 : T] : y^*_t = O\}$.

Our **baseline** method is simply to evaluate the classifier performance based on y^*. As we mentioned above, our specific context for probability statement extraction leads us to discard all labels in a sentence that does not contain any P label. We thus implement this constraint in our baseline as a simple post-processing filter on the CRF output. Specifically, for probability statement extraction, for a sentence such that $|t_P| = 0$, then we define a modified output y^B (B standing here for Baseline) where $y^B_t = O$ $\forall t = 1 : T$, else we keep the original output so that $y^B_t = y^*_t$ $\forall t = 1 : T$.

3.2 CRF-Driven Constraints

Our chosen method to enforce constraints takes into account current knowledge (i.e., initial decoding from CRF) when estimating probabilities of labels by conditioning posterior probabilities on the presence of label A and B based on current observations. Specifically, if only A missing, i.e.,

$|t_A| = 0$ then we search

$$t^*_A = \underset{t \in t_O}{\operatorname{argmax}} \, \varphi_t(s) \quad (1)$$

and we define $y^C_t = A \; \forall t \in t^*_A$ and $y^C_t = y^B_t \; \forall t \notin t^*_A$.

In this method, φ represents the the posterior *conditional* probability given the observed locations of the required labels. In the case of probability statement extraction, we have: $\varphi_t(s) = \Pr(y_t = s | x, y_{t_P} = P, y_{t_A} = A, y_{t_B} = B)$ for $s \in \{A, B, P, O\}$. In the case of dependence relation extraction, we have: $\varphi_t(s) = \Pr(y_t = s | x, y_{t_I} = I, y_{t_A} = A, y_{t_B} = B)$ for $s \in \{A, B, I, O\}$.

Similar procedures are applied when B is missing and when both labels are missing where we search jointly on the best locations for A and B.

One difficulty with this approach is the need to compute φ. For this purpose, we borrow from Culotta and McCallum (2004) who provide a method to compute the forward values $\alpha_t(s) = \Pr(x_1, \ldots, x_t, y_t = s)$ of the forward-backward algorithm when forced to conform to a subpath of constraints $C = \langle s_t, s_{t+1}, \ldots \rangle$. These constraints specify for a subset of locations which states they must be in or not be in (negative constraints).

The original recursive approach to compute $\alpha_t(s)$ is

$$\alpha_{t+1}(s) = \sum_{s'} \alpha_t(s') \exp\left(\sum_k \lambda_k f_k(s', s, x_{t+1})\right) \quad (2)$$

The updated recursion equation proposed by Culotta and McCallum (2004) so as to compute $\alpha'_t(s) = \Pr(x_1, \ldots, x_t, y_t = s | C)$ is simply to apply Equation 2 when $y_{t+1} = s$ conforms with C (including locations that are not constrained in any way in C) and set $\alpha'_{t+1}(s) = 0$ otherwise. Note that in (Culotta and McCallum, 2004), the indexes of the constraints included in C are assumed to be contiguous although the method also applies when they are not. In our case, for probability statement extraction, we will simply set $C = \{y_{t_P} = P, y_{t_A} = A, y_{t_B} = B\}$, again where at least one of the sets t_A or t_B is empty.

We similarly extend this method to compute constrained backward values $\beta'_t(s) = \Pr(x_{t+1}, \ldots, x_T | y_t = s, C)$ by proposing the modified backward recursion

$$\beta'_t(s) = \sum_{s'} \beta'_{t+1}(s') \exp\left(\sum_k \lambda_k f_k(s, s', x_{t+1})\right) \quad (3)$$

when $y_t = s$ conforms with C and set $\beta'_{t+1}(s) = 0$ otherwise. Overall, this means that $\varphi_t(s) = \Pr(y_t = s | x, y_{t_P} = P, y_{t_A} = A, y_{t_B} = B)$ can be expressed as follows:

$$\varphi_t(s) = \frac{\Pr(y_t = s, x | y_{t_P} = P, y_{t_A} = A, y_{t_B} = B)}{\Pr(x | y_{t_P} = P, y_{t_A} = A, y_{t_B} = B)} \quad (4)$$

This is the result of an application of Bayes rule along with the conditional independence assumptions of the CRF. In turn, we have that :

$$\varphi_t(s) = \frac{\alpha'_t(s)\beta'_t(s)}{\sum_{s'} \alpha'_T(s')} \quad (5)$$

The output y^C is guaranteed to contain at least one label A and one label B. t^*_A in Equation 1 will contain only one token while variables A and B, in reality, often span multiple tokens. Therefore, as an additional post-processing step, we run the Viterbi algorithm once more using the identified labels A, B and P as constraints. This may reveal longer spans with such labels.

3.3 Classifier-driven Constraints

As we report in Section 5, imposing constraints based on the initial CRF decoding improves recall more than it degrades precision and thus proves useful. We further explore whether adding flexibility to the process can help reduce the precision degradation. Indeed in the case of dependence relations, we observed that several influence terms are quite common in dependence relations but are still not found exclusively in these relations. In our dataset for instance, the most common influence term words are *associated* (142 occurrences in the training set), *association* (42), *risk* (36), and *increased* (22). Our one-step CRF-based approach may thus be misled by those common words into forcing the presence of a dependence relation.

Therefore, we introduce two separate binary classifiers to predict whether or not the sentence contains either a dependence relation or a probability statement. Our intuition is that the classifier will be a more reliable indicator of the presence of a relation than the entity extraction of either probability number or influence term. We make use of that prediction to determine if and how to apply the constraints. In fact, we apply a threshold on the confidence of the classifier in order to decide whether to enforce the constraints.

To coordinate the classification, the entity detection (baseline CRF) and the constraint enforce-

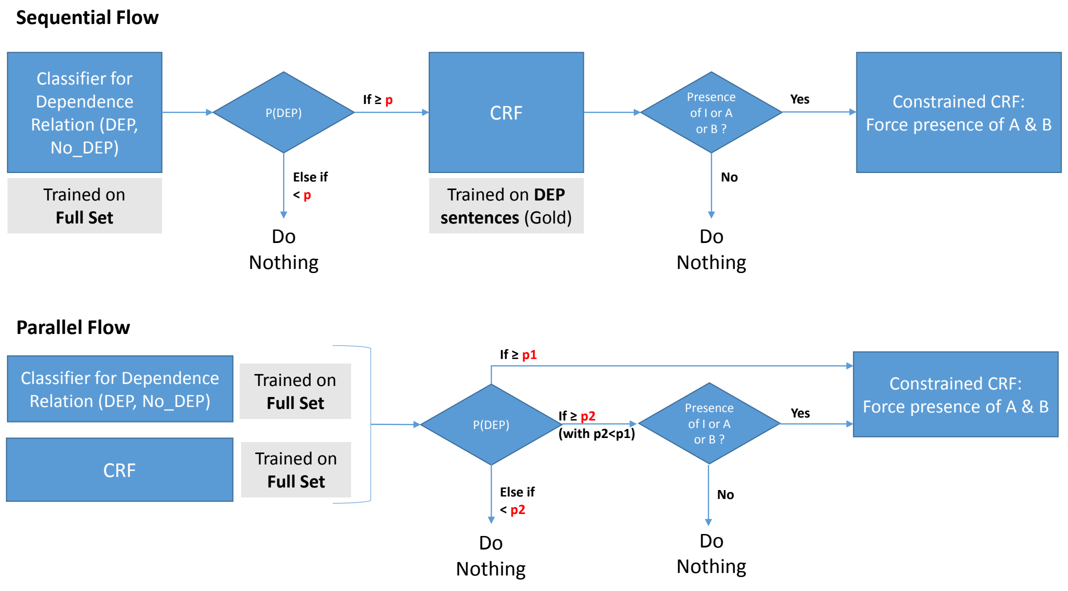
Sequential Flow
Classifier for Dependence Relation (DEP, No_DEP)
Trained on Full Set
P(DEP)
If ≥ p
Else if < p
Do Nothing
CRF
Trained on DEP sentences (Gold)
Presence of I or A or B ?
Yes
No
Do Nothing
Constrained CRF: Force presence of A & B
Parallel Flow
Classifier for Dependence Relation (DEP, No_DEP)
Trained on Full Set
CRF
Trained on Full Set
If ≥ p1
P(DEP)
If ≥ p2 (with p2<p1)
Else if < p2
Do Nothing
Presence of I or A or B ?
Yes
No
Do Nothing
Constrained CRF: Force presence of A & B

Figure 2: Description of Constraint Enforcement Flows

ment steps, we explore two kinds of flows as depicted in Figure 2 for dependence relation extraction. Both flows start with the binary classifier but evolve differently:

- In the *sequential flow*, if the confidence of the classifier is below a threshold p we simply discard the sentence. If it is above the threshold, then we process the sentence through the baseline CRF trained only on sentences containing relations in the ground truth. We then enforce our constraints, i.e., for the dependence case, discarding the sentence if no variable or influence term has been found but forcing the presence of variables if we detect an influence term or at least one variable.

- In the *parallel flow*, we implement the baseline CRF in parallel with the binary classifier. This implies that the baseline CRF is trained on the full set of sentences. Afterwards, if the confidence of the classifier is below p^2, we discard the sentence, if it is above p^1 we enforce our constraints regardless of the Baseline CRF output. For intermediate cases, we only enforce constraints if we detect either an influence term or a variable.

4 Experiments

In this section we first describe the data used in our experiments and then cover the configuration of the baseline CRF classifier along with the constrained configurations we test.

Data for Dependence Statements. The dependence dataset comes from 210 abstracts selected from PubMed based on a query about breast cancer. These 210 abstracts are split into 2144 sentences, of which 785 have a dependence relation. The dependence relations include 830 variables labeled A and 837 labeled B. The annotation also includes labels for *influence terms*, *modifiers* and *negation* (as mentioned in Section 2.1). The latter two are not considered here as they do not contribute to the dependence relations, but we do show results for influence terms as they are useful in constructing the dependence relations (although that is not covered in this paper).

Data for Probability Statements. The probability dataset is similarly constructed with 194 abstracts from PubMed that are related to breast cancer. The whole dataset has 2078 sentences, 376 of which, contain probabilistic statements, for example, "$\underline{81\%}_{\mathbf{P}}$ of $\underline{\text{stage III/IV breast cancers}}_{\mathbf{B}}$ were positive for $\underline{\text{SNCG expression}}_{\mathbf{A}}$." These sentences contain 652 probability terms, 446 variables A, and 467 variables B.

We split both datasets into train (70%) and test (30%) sets. Parameter tuning can be done by splitting the training set, however experiments shown in this paper do not require parameter tuning.

Baseline CRF We initially evaluate a baseline CRF model without constraints, implemented with Mallet (McCallum, 2002). The same feature set used in the baseline is used throughout our experiments. It is composed of standard features for sequence tagging: surface form, lemma, POS tag, word shape (i.e., is capitalized, has digit, etc.), and the arc label from a dependency parse. The CRF also extracts features from the previous position ($t-1$) and the following position ($t+1$) as well as bigram features combining the previous two positions ($t-2, t-1$). We do not tune the features or regularization parameters in our experiments, but instead focus on the differences from applying constraints in decoding. The CRF training is the same for nearly all experiments and the important changes are in decoding and in how the constraints are applied, so tuning these parameters should not affect our results. The one exception to this is the sequential flow where the CRF classifier is trained only on sentences with relations. We use the default value for the Gaussian variance prior (10) and keep the same features across our experiments.

In addition to the baseline CRF we test three constraint settings: (i) **Default** refers to the decoding with the CRF driven enforcement of constraints described in Section 3.2; (ii) **Parallel** refers to the application of the Parallel flow which feeds the sentences into the CRF to identify entities and a binary classifier to determine if the sentence has a relation. The output of the binary classifier determines if and how the decoding constraints should be applied; and (iii) **Sequential** refers to the application of the sequential flow where the binary classifier first filters out noisy sentences followed by the use of a CRF trained on relation sentence data.

For the binary classifier, we make use of IBM's Natural Language Classifier service[3] which is relies on a Convolutional Neural Network combined with word embeddings (Feng et al., 2015). To clarify, the classifiers that we use, while using pre-trained word embedding (on general domain), are then only trained with our own training data.

Evaluation criteria. We evaluate our approach using standard metrics: precision, recall, and F_1. In addition, we consider different criteria for matching entities, i.e., token matches, exact entity matches, and "sloppy" matches (Olsson et al., 2002). In Section 5 we report only our results for token matching since the trends are consistent for different matching criteria. Exact entity matching is unnecessarily strict for our application (e.g., in the context of our work "AC haplotype" is just as good as "Carriers of the AC haplotype") and measuring performance by token makes the results easily interpretable.

5 Results

5.1 Binary Classifier

As mentioned in Section 3.3, we introduce a binary classifier to decide whether constraints should be applied or not (instead of relying only on the entity annotation of variables A, B, and Influence or Probability terms). We need an accurate classifier to ensure that constraints are only applied when necessary. Our experiments show that the classifier achieves good results with accuracy for classifying dependence sentences of 82.2% and for probability statements, 95.3%.

5.2 Entity Extraction for Dependence Relations

We first look at the entity extraction results for dependence relations in Table 1. There is consistent improvement in F_1 scores for variables A and B as we refine the application of constraints. The baseline CRF classifier has modest precision but much weaker recall and misses a number of variable A and B entities entirely. We are not particularly concerned about boundary errors with the CRF and they are relatively infrequent. On the other hand, the missed entities, i.e., the entity *false negatives*, are more detrimental to the results. In fact, of the 606 sentences in the test set, 253 should have some dependence relation annotation but 143 of these are missing a variable A, 143 are missing a variable B, and 84 are missing both. This result motivates our use of domain constraints.

Adding Default constraints leads to an increase in F_1 as recall improves while precision drops. When the baseline CRF assigns any A, B, or influence term, we force it to have both variables A and B, and essentially by forcing it we recover enough new variables (higher recall) to offset making some less confident prediction (lower precision). The application of these constraints depends on the quality of the Baseline predictions for A, B, and Influence Term, and we note that that

[3]https://www.ibm.com/watson/developercloud/nl-classifier.html

	Precision	Recall	$F_{\beta=1}$
VarA			
Baseline	60.27%	34.26%	43.69
Default	55.34%	45.68%	50.05
Parallel	**64.16%**	44.95%	52.87
Sequential	56.86%	**62.19%**	**59.41**
VarB			
Baseline	41.53%	24.54%	30.85
Default	43.19%	40.44%	41.77
Parallel	**52.15%**	40.11%	45.35
Sequential	46.78%	**48.67%**	**47.71**
Influence Term			
Baseline	**68.45%**	42.95%	52.78
Default	67.18%	43.96%	53.14
Parallel	65.10%	55.70%	60.04
Sequential	64.86%	**56.38%**	**60.32**

Table 1: Entity extraction results for dependence relations.

even for influence terms our performance is moderate and inferior to that of the binary classifier.

The Parallel results display improvements to both precision and recall over the baseline. While Parallel recall for A and B is slightly below the numbers reached in Default, the 9% absolute increase in precision for both variables A and B leads to F_1 improvements for both. By applying the binary classifier prediction before applying constraints, we remove some noisy sentences that contain spuriously labeled entities, which are not in a dependence relation; in the Parallel results there are 56 such sentences. For example, in the baseline experiment a lone variable B, *subsequent cancer occurrence*, is extracted from the sentence "It allowed for a correct estimation of the risk, and for investigating the time trend of the subsequent cancer occurrence." The classifier correctly filters out this sentence and prevents the CRF from classifying what might be a valid variable B had there actually been a dependence relation.

Naturally, by filtering out sentences we also risk discarding some with dependence relations and thus affecting recall. In one example of this negative case, the baseline experiment labels only an influence term without variables A or B. The constraints applied for the Default constraint experiment assign the (correct) variable B and (incorrect) variable A. However, the classifier in the Parallel experiment mistakenly leads us not to label any entities in this sentence, thereby missing the relation.

We observe similar patterns for the Sequential results. The overall best F_1 scores for variables A and B are due to the significant increases in recall. The main difference between Parallel and Sequential is the fact that the CRF is trained on a filtered set of sentences for Sequential compared to the full set for Parallel. The Sequential CRF, using filtered data, has a higher ratio of variables labeled A and B per word seen in training and this makes it more confident in predicting labels A and B in testing. This helps the Sequential flow recover several new entities which are missed by the baseline CRF. This also explains the higher recall but lower precision with respect to Parallel.

Although only variables A and B are necessary for dependence relations, we also report performance on influence terms, which contribute to the identification of dependence relations. For the influence terms, we observe that precision decreases with each experiment as recall increases. The binary classifier has a strong and positive effect on influence term extraction as we see an increase in recall and F_1. Recall for Parallel and Sequential increases by about 11% and 15% (absolute).

Table 1 shows Parallel results with $p^1 = 0.5$ and $p^2 = 0.5$ and Sequential results with $p = 0.5$ (see Figure 2). We chose those parameter values as they correspond to the default for a binary classifier. In Section 5.4 we look more closely at the effects of p, p^1 and p^2.

5.3 Entity Extraction for Probability Statement

The probability results reported in Table 2 display similar trends as those for dependence relations, though we note a few differences as well. First the Baseline results are lower for variables A and B. This is due mainly to the heterogeneity of these variables (even with respect to the dependence relations) and to the smaller dataset. To illustrate how much more heterogeneous the variables are, the type-token ratio for variable A is 0.188 in the dependence training set and 0.392 in the probability training set, and the difference for variable B is similar, 0.224 vs. 0.448. The number of sentences in the dependence and probability corpora are similar but probability statements are less frequent than dependence relations and for the 620

	Precision	Recall	$F_{\beta=1}$
VarA			
Baseline	33.98%	8.93%	14.14
Default	52.81%	31.12%	39.17
Parallel	53.41%	33.93%	41.50
Sequential	**57.41%**	**46.43%**	**51.34**
VarB			
Baseline	38.61%	11.40%	17.61
Default	43.27%	26.32%	32.73
Parallel	39.01%	25.44%	30.80
Sequential	**47.73%**	**42.98%**	**45.23**
Probability Term			
Baseline	79.43%	69.24%	73.99
Default	79.65%	70.17%	74.61
Parallel	**81.56%**	69.71%	75.17
Sequential	80.36%	**75.89%**	**78.06**

Table 2: Entity extraction results for probability statements.

	Dependence		Probability	
p	VarA	VarB	VarA	VarB
0.00	48.94	36.23	51.10	44.70
0.05	57.01	46.69	**52.40**	**47.22**
0.25	**59.77**	47.05	51.25	46.90
0.50	59.41	47.71	51.34	45.23
0.75	59.41	49.42	51.28	45.55
0.95	58.72	**49.52**	46.37	39.79
1.00	0.00	0.00	0.00	0.00

Table 3: F_1 scores across p threshold values for Sequential flow.

sentences in the test set we only have 127 with probability annotation. Like with dependence relations, for probability statements we can motivate our constraints by looking at the large number of missed (i.e., false negative) entities. The baseline CRF misses variable A in 94 sentences with probability statements (i.e., about 74% of probability statements are missing variable A). There are 78 probability sentences with a missed variable B, and 57 missing both.

By applying constraints to these missing variables we observe the largest increase in performance occurs from the Baseline to the Default setting, with improvements in precision, recall, and F_1. The Default constraints approach hinges on the CRF prediction of the probability term, which performs well. Because this prediction is already reliable, by contrast with the dependence relation extraction, there is less potential for improvement by applying the binary classifier. In fact, for Parallel, performance increases for variable A but decreases for variable B.

Training the CRF solely on probability statements, as is the case in Sequential, appears to have a greater impact than with dependence. As the proportion of sentences with no-relation in the probability dataset is higher than in the dependence dataset, filtering out the no-relations sentences removes more noise in the case of probability statement extraction. In turn, removing this noise improves the CRF performance, leading in particular to the improved performance for the probability term and consequently to improvements in the variable extraction with constraints.

5.4 Improving Performance through Threshold Values

Performance was earlier reported for the Parallel and Sequential flow for $p = p^1 = p^2 = 0.5$, which represents the default threshold for binary classifiers. In this section, we look closer at the effect of these threshold values on performance.

There are no values for p that consistently lead to maximum F_1 scores (Tables 3–7) and there is not space to show precision and recall results as well. However, as would be expected, the precision goes up as p^1 and p^2 increase, i.e., as we become more conservative in constraining the CRF output. On the other hand, recall drops with higher values of the p threshold as the CRF is not pushed to find previously missed variables A or B.

We do find lower p^2 thresholds perform better for probability than dependence. This is likely because the probability constraints based on the CRF are still quite reliable. The classifier for dependence statements must be more strict in filtering sentences, using higher p^2 thresholds, because the dependence constraints are less reliable.

The Sequential flow results are similar. The maximum F_1 scores on the probability dataset come with $p = 0.05$, like p^2 values for Parallel. The p threshold for variables A and B varies, also similar to p^2 for Parallel, with better A results coming from a lower threshold and better B results with a higher threshold. However, more work needs to be done to see how we can best leverage classification in the Sequential and Parallel flows.

p^2 / p^1	0.00	0.05	0.25	0.50	0.75	0.95
0.00	46.48	–	–	–	–	–
0.05	50.81	52.64	–	–	–	–
0.25	50.87	52.34	53.88	–	–	–
0.50	51.26	52.77	**53.93**	52.87	–	–
0.75	51.18	52.69	53.85	52.34	52.30	–
0.95	51.17	52.66	53.83	52.32	51.82	51.33
1.00	50.05	51.53	52.70	51.10	50.55	49.67

Table 4: Var. A F_1 for Dependence Parallel flow.

p^2 / p^1	0.00	0.05	0.25	0.50	0.75	0.95
0.00	36.98	–	–	–	–	–
0.05	42.42	44.93	–	–	–	–
0.25	43.38	44.81	45.34	–	–	–
0.50	43.36	44.80	44.61	45.35	–	–
0.75	43.10	44.53	44.33	44.67	**45.45**	–
0.95	43.18	44.61	44.41	44.50	44.79	44.78
1.00	41.77	43.13	42.86	42.91	43.18	42.66

Table 5: Var. B F_1 for Dependence Parallel flow.

p^2 / p^1	0.00	0.05	0.25	0.50	0.75	0.95
0.00	18.88	–	–	–	–	–
0.05	42.50	**42.75**	–	–	–	–
0.25	41.01	41.22	41.35	–	–	–
0.50	41.38	41.60	41.73	41.50	–	–
0.75	40.90	41.11	41.24	41.00	41.01	–
0.95	40.74	40.94	41.07	40.83	40.84	34.49
1.00	39.17	39.34	39.47	39.20	39.19	32.35

Table 6: Var. A F_1 for Probability Parallel flow.

p^2 / p^1	0.00	0.05	0.25	0.50	0.75	0.95
0.00	13.90	–	–	–	–	–
0.05	31.39	31.58	–	–	–	–
0.25	31.31	31.51	31.27	–	–	–
0.50	31.85	32.06	31.82	30.80	–	–
0.75	31.96	32.17	31.93	30.91	30.88	–
0.95	31.87	32.09	31.84	30.80	30.77	21.78
1.00	32.73	**32.96**	32.71	31.64	31.62	22.31

Table 7: Var. B F_1 for Probability Parallel flow.

6 Related Work

Our work touches on several areas from constrained conditional models (Goldwasser et al., 2012) to biomedical entity extraction. The work most related to our approach for applying constraints with CRF decoding is (Culotta and McCallum, 2004; Roth and Yih, 2005; Kristjansson et al., 2004). Our solution for constraining CRF decoding borrows from Culotta and McCallum (2004) (which is also used by Kristjansson et al. (2004)). They use constraints for calculating the 'forward' values in the forward-backward algorithm and use this for estimating confidence at given states in the sequence. Our work applies constraints both forward and backward and uses these global constraints to force specific entities to be extracted. Roth and Yih (2005) also apply constraints to the CRF but instead of using Viterbi decoding as is done here they use Integer Linear Programming (ILP) to add constraints in decoding for semantic role labeling.

With respect to our overall objective of entity and relation extraction from the medical literature, a large proportion of the related work originates from BioNLP event extraction (Kim et al., 2009; Nédellec et al., 2013; Chaix et al., 2016; Deléger et al., 2016). These tasks are similar in that they extract biomedical entities, analogous to our variables A and B, and the relations between entities (e.g., bio-molecular events).

The most similar entity extraction task to ours is from Fiszman et al. (2007). They are interested in extracting mentions of diseases and medical risk factors[4] from medical literature. They take a less statistical and more semantic approach to convert the biomedical text into a semantic representation using the UMLS Semantic Network.

7 Conclusions

In this paper we investigate how we can improve performance on information extraction tasks by constraining CRF-based approaches. We investigate two relation extraction tasks from the medical literature – dependence relations and probability statements – and show that by using our constrained CRF models we can get significant improvements over a CRF baseline. In future work we plan to build on these improvements and test constraints jointly applied to entity and relation extraction to improve our project's construction of decision support models.

[4]Their *risks* and *disorders* appear to be subsets of variables A and B.

References

Estelle Chaix, Bertrand Dubreucq, Abdelhak Fatihi, Dialekti Valsamou, Robert Bossy, Mouhamadou Ba, Louise Deléger, Pierre Zweigenbaum, Philippe Bessières, Loïc Lepiniec, and Claire Nédellec. 2016. Overview of the regulatory network of plant seed development (seedev) task at the bionlp shared task 2016. In *Proceedings of the 4th BioNLP Shared Task Workshop*, pages 1–11, Berlin, Germany, August. Association for Computational Linguistics.

Aron Culotta and Andrew McCallum. 2004. Confidence estimation for information extraction. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Short Papers*, pages 109–112, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Louise Deléger, Robert Bossy, Estelle Chaix, Mouhamadou Ba, Arnaud Ferré, Philippe Bessières, and Claire Nédellec. 2016. Overview of the bacteria biotope task at bionlp shared task 2016. In *Proceedings of the 4th BioNLP Shared Task Workshop*, pages 12–22, Berlin, Germany, August. Association for Computational Linguistics.

Léa A. Deleris and Charles Jochim. 2016. Probability statements extraction with constrained conditional random fields. In *Proceedings of MIE2016*, pages 527–531.

Léa Amandine Deleris, Bogdan Sacaleanu, and Lamia Tounsi. 2013. Extracting risk modeling information from medical articles. In *MEDINFO 2013 - Proceedings of the 14th World Congress on Medical and Health Informatics, 20-13 August 2013, Copenhagen, Denmark*, page 1158.

Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2015, Scottsdale, AZ, USA, December 13-17, 2015*, pages 813–820.

Marcelo Fiszman, Graciela Rosemblat, Caroline B. Ahlers, and Thomas C. Rindflesch. 2007. Identifying risk factors for metabolic syndrome in biomedical text. In *Proceedings of the AMIA Annual Symposium*, pages 249–253.

Dan Goldwasser, Vivek Srikumar, and Dan Roth. 2012. Predicting structures in nlp: Constrained conditional models and integer linear programming in nlp. In *NAACL HLT 2012 Tutorial Abstracts*, Montréal, Canada, June. Association for Computational Linguistics.

Dimitar Hristovski, Carol Friedman, Thomas C. Rindflesch, and Borut Peterlin. 2006. Exploiting semantic relations for literature-based discovery. In *AMIA Annual Symposium Proceedings*, pages 349–353.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. 2009. Overview of BioNLP'09 shared task on event extraction. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 1–9, Boulder, Colorado, June. Association for Computational Linguistics.

Trausti Kristjansson, Aron Culotta, Paul Viola, and Andrew McCallum. 2004. Interactive information extraction with constrained conditional random fields. In *Proceedings of the 19th National Conference on Artifical Intelligence*, AAAI'04, pages 412–418. AAAI Press.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

Claire Nédellec, Robert Bossy, Jin-Dong Kim, Jung-Jae Kim, Tomoko Ohta, Sampo Pyysalo, and Pierre Zweigenbaum. 2013. Overview of BioNLP shared task 2013. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 1–7, Sofia, Bulgaria, August. Association for Computational Linguistics.

Fredrik Olsson, Gunnar Eriksson, Kristofer Franzén, Lars Asker, and Per Lidén. 2002. Notions of correctness when evaluating protein name taggers. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dan Roth and Wen-tau Yih. 2005. Integer linear programming inference for conditional random fields. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, pages 736–743, New York, NY, USA. ACM.

Burr Settles. 2004. Biomedical named entity recognition using conditional random fields and rich feature sets. In Nigel Collier, Patrick Ruch, and Adeline Nazarenko, editors, *COLING 2004 International Joint workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA/BioNLP) 2004*, pages 107–110, Geneva, Switzerland, August 28th and 29th. COLING.

Fei Sha and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 134–141, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for nlp-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107, Avignon, France, April. Association for Computational Linguistics.