Revealing the Structure of Medical Dictations with Conditional Random Fields

Jeremy Jancsary and **Johannes Matiasek**
Austrian Research Institute for Artificial Intelligence
A-1010 Vienna, Freyung 6/6
firstname.lastname@ofai.at

Harald Trost
Department of Medical Cybernetics and Artificial Intelligence
of the Center for Brain Research, Medical University Vienna, Austria
harald.trost@meduniwien.ac.at

Abstract

Automatic processing of medical dictations poses a significant challenge. We approach the problem by introducing a statistical framework capable of identifying types and boundaries of sections, lists and other structures occurring in a dictation, thereby gaining explicit knowledge about the function of such elements. Training data is created semi-automatically by aligning a parallel corpus of corrected medical reports and corresponding transcripts generated via automatic speech recognition. We highlight the properties of our statistical framework, which is based on conditional random fields (CRFs) and implemented as an efficient, publicly available toolkit. Finally, we show that our approach is effective both under ideal conditions and for real-life dictation involving speech recognition errors and speech-related phenomena such as hesitation and repetitions.

1 Introduction

It is quite common to dictate reports and leave the typing to typists – especially for the medical domain, where every consultation or treatment has to be documented. Automatic Speech Recognition (ASR) can support professional typists in their work by providing a transcript of what has been dictated. However, manual corrections are still needed. In particular, speech recognition errors have to be corrected. Furthermore, speaker errors, such as hesitations or repetitions, and instructions to the transcriptionist have to be removed. Finally, and most notably, proper structuring and formatting of the report has to be performed. For the medical domain, fairly clear guidelines exist with regard to what has to be dictated, and how it should be arranged. Thus, missing headings may have to be inserted, sentences must be grouped into paragraphs in a meaningful way, enumeration lists may have to be introduced, and so on.

The goal of the work presented here was to ease the job of the typist by formatting the dictation according to its structure and the formatting guidelines. The prerequisite for this task is the identification of the various structural elements in the dictation which will be be described in this paper.

```
complaint dehydration weakness and diarrhea
full stop Mr.  Will Shawn is a 81-year-old
cold Asian gentleman who came in with fever
and Persian diaper was sent to the emergency
department by his primary care physician due
him being dehydrated period ... neck physical
exam general alert and oriented times three
known acute distress vital signs are stable
... diagnosis is one chronic diarrhea with
hydration he also has hypokalemia neck number
thromboctopenia probably duty liver cirrhosis
... a plan was discussed with patient in
detail will transfer him to a nurse and
facility for further care ... end of dictation
```

Fig. 1: Raw output of speech recognition

Figure 1 shows a fragment of a typical report as recognized by ASR, exemplifying some of the problems we have to deal with:

- Punctuation and enumeration markers may be dictated or not, thus sentence boundaries and numbered items often have to be inferred;
- the same holds for (sub)section headings;
- finally, recognition errors complicate the task.

Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, pages 1–10,
Honolulu, October 2008.

```
CHIEF COMPLAINT
Dehydration, weakness and diarrhea.

HISTORY OF PRESENT ILLNESS
Mr.  Wilson is a 81-year-old Caucasian
gentleman who came in here with fever and
persistent diarrhea.  He was sent to the
emergency department by his primary care
physician due to him being dehydrated.
...

PHYSICAL EXAMINATION
  GENERAL: He is alert and oriented times
    three, not in acute distress.

  VITAL SIGNS: Stable.
...

DIAGNOSIS
  1. Chronic diarrhea with dehydration.  He
    also has hypokalemia.

  2. Thromboctopenia, probably due to liver
    cirrhosis.
...

PLAN AND DISCUSSION
The plan was discussed with the patient
in detail.  Will transfer him to a nursing
facility for further care.
...
```

Fig. 2: A typical medical report

When properly edited and formatted, the same dictation appears significantly more comprehensible, as can be seen in figure 2. In order to arrive at this result it is necessary to identify the inherent structure of the dictation, i.e. the various hierarchically nested segments. We will recast the segmentation problem as a multi-tiered tagging problem and show that indeed a good deal of the structure of medical dictations can be revealed.

The main contributions of our paper are as follows: First, we introduce a generic approach that can be integrated seamlessly with existing ASR solutions and provides structured output for medical dictations. Second, we provide a freely available toolkit for factorial conditional random fields (CRFs) that forms the basis of aforementioned approach and is also applicable to numerous other problems (see section 6).

2 Related Work

The structure recognition problem dealt with here is closely related to the field of *linear text segmentation* with the goal to partition text into coherent blocks, but on a single level. Thus, our task generalizes linear text segmentation to multiple levels.

A meanwhile classic approach towards domain-independent linear text segmentation, *C99*, is presented in Choi (2000). *C99* is the baseline which many current algorithms are compared to. Choi's algorithm surpasses previous work by Hearst (1997), who proposed the *Texttiling* algorithm. The best results published to date are – to the best of our knowledge – those of Lamprier et al. (2008).

The automatic detection of (sub)section *topics* plays an important role in our work, since changes of topic indicate a section boundary and appropriate headings can be derived from the section type. Topic detection is usually performed using methods similar to those of *text classification* (see Sebastiani (2002) for a survey).

Matsuov (2003) presents a dynamic programming algorithm capable of segmenting medical reports into sections and assigning topics to them. Thus, the aims of his work are similar to ours. However, he is not concerned with the more fine-grained elements, and also uses a different machinery.

When dealing with tagging problems, statistical frameworks such as HMMs (Rabiner, 1989) or, recently, CRFs (Lafferty et al., 2001) are most commonly applied. Whereas HMMs are generative models, CRFs are discriminative models that can incorporate rich features. However, other approaches to text segmentation have also been pursued. E.g., McDonald et al. (2005) present a model based on multilabel classification, allowing for natural handling of overlapping or non-contiguous segments.

Finally, the work of Ye and Viola (2004) bears similarities to ours. They apply CRFs to the parsing of hierarchical lists and outlines in handwritten notes, and thus have the same goal of finding deep structure using the same probabilistic framework.

3 Problem Representation

For representing our segmentation problem we use a trick that is well-known from chunking and named entity recognition, and recast the problem as a tagging problem in the so-called `BIO`[1] notation. Since we want to assign a type to every segment, OUTSIDE labels are not needed. However, we perform seg-

[1] BEGIN - INSIDE - OUTSIDE

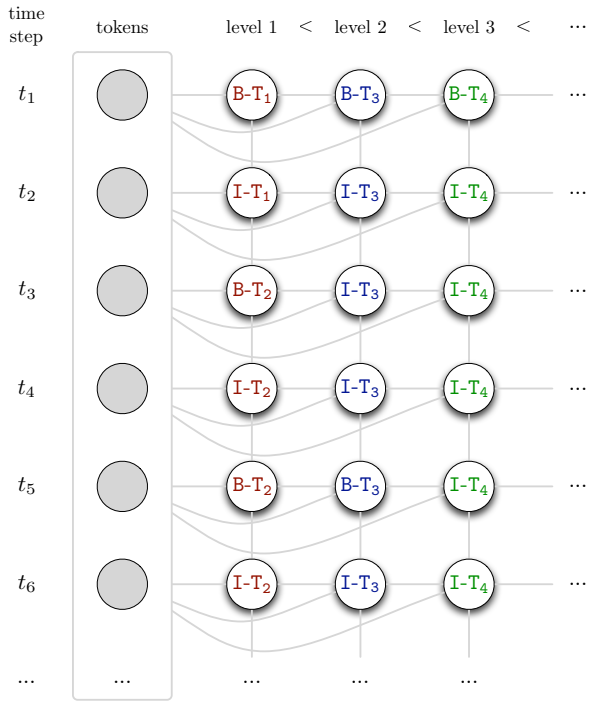

Fig. 3: Multi-level segmentation as tagging problem

mentation on multiple levels, therefore multiple *label chains* are required. Furthermore, we also want to assign *types* to certain segments, thus the labels need an encoding for the type of segment they represent. Figure 3 illustrates this representation: `B-T`$_i$ denotes the beginning of a segment of type `T`$_i$, while `I-T`$_i$ indicates that the segment of type `T`$_i$ continues. By adding label chains, it is possible to group the segments of the previous chain into coarser units. Tree-like structures of unlimited depth can be expressed this way[2]. The gray lines in figure 3 denote dependencies between nodes. Node labels also depend on the input token sequence in an arbitrarily wide context window.

4 Data Preparation

The raw data available to us consists of two parallel corpora of 2007 reports from the area of medical consultations, dictated by physicians. The first corpus, C_{RCG}, consists of the raw output of ASR (figure 1), the other one, C_{COR}, contains the corresponding corrected and formatted reports (figure 2).

In order to arrive at an annotated corpus in a format suitable for the tagging problem, we first have to analyze the report structure and define appropriate labels for each segmentation level. Then, every token has to be annotated with the appropriate begin or inside labels. A report has 625 tokens on average, so the manual annotation of roughly 1.25 million tokens seemed not to be feasible. Thus we decided to produce the annotations programmatically and restrict manual work to corrections.

4.1 Analysis of report structure

When inspecting reports in C_{COR}, a human reader can easily identify the various elements a report consists of, such as *headings* – written in bold on a separate line – introducing sections, *subheadings* – written in bold followed by a colon – introducing subsections, and *enumerations* starting with indented numbers followed by a period. Going down further, there are *paragraphs* divided into *sentences*. Using these structuring elements, a hierarchic data structure comprising all report elements can be induced.

Sections and subsections are typed according to their heading. There exist clear recommendations on structuring medical reports, such as E2184-02 (ASTM International, 2002). However, actual medical reports still vary greatly with regard to their structure. Using the aforementioned standard, we assigned the (sub)headings that actually appeared in the data to the closest type, introducing new types only when absolutely necessary. Finally we arrived at a structure model with three label chains:

- **Sentence level**, with 4 labels: `Heading`, `Subheading`, `Sentence`, `Enummarker`
- **Subsection level**, with 45 labels: `Paragraph`, `Enumelement`, `None` and 42 subsection types (e.g. `VitalSigns`, `Cardiovascular` ...)
- **Section level**, with 23 section types (e.g. `ReasonForEncounter`, `Findings`, `Plan` ...)

4.2 Corpus annotation

Since the reports in C_{COR} are manually edited they are reliable to parse. We employed a broad-coverage dictionary (handling also multi-word terms) and a domain-specific grammar for parsing and layout information. A regular heading grammar was used for mapping (sub)headings to the defined (sub)section labels (for details see Jancsary (2008)). The output

[2]Note, that since we omit a redundant top-level chain, this structure technically is a *hedge* rather than a *tree*.

C_{COR}		OP	C_{RCG}	
...	...	...	...	...
B − Head	CHIEF	del		
Head	COMPLAINT	sub	complaint	B − Head
B − Sent	Dehydration	sub	dehydration	B − Sent
Sent	,	del		
Sent	weakness	sub	weakness	Sent
Sent	and	sub	and	Sent
Sent	diarrhea	sub	diarrhea	Sent
Sent	.	sub	fullstop	Sent
B − Sent	Mr.	sub	Mr.	B − Sent
Sent	Wilson	sub	Will	Sent
		ins	Shawn	Sent
Sent	is	sub	is	Sent
Sent	a	sub	a	Sent
Sent	81-year-old	sub	81-year-old	Sent
Sent	Caucasian	sub	cold	Sent
Sent		ins	Asian	Sent
Sent	gentleman	sub	gentleman	Sent
Sent	who	sub	who	Sent
Sent	came	sub	came	Sent
Sent	in	del		
Sent	here	sub	here	Sent
Sent	with	sub	with	Sent
Sent	fever	sub	fever	Sent
Sent	and	sub	and	Sent
Sent	persistent	sub	Persian	Sent
Sent	diarrhea	sub	diaper	Sent
Sent	.	del		
...	...	...	...	...

Fig. 4: Mapping labels via alignment

of the parser is a hedge data structure from which the annotation labels can be derived easily.

However, our goal is to develop a model for recognizing the report structure from the *dictation*, thus we have to map the newly created annotation of reports in C_{COR} onto the corresponding reports in C_{RCG}. The basic idea here is to align the tokens of C_{COR} with the tokens in C_{RCG} and to copy the annotations (cf. figure 4[3]). There are some peculiarities we have to take care of during alignment:

1. non-dictated items in C_{COR} (e.g. punctuation, headings)
2. dictated words that do not occur in C_{COR} (meta instructions, repetitions)
3. non-identical but corresponding items (recognition errors, reformulations)

Since it is particularly necessary to correctly align items of the third group, standard string-edit distance based methods (Levenshtein, 1966) need to be augmented. Therefore we use a more sophisticated cost function. It assigns tokens that are similar (either from a semantic or phonetic point of view) a low cost for substitution, whereas dissimilar tokens receive a prohibitively expensive score. Costs for deletion and insertion are assigned inversely. Semantic similarity is computed using Wordnet (Fellbaum, 1998) and UMLS (Lindberg et al., 1993). For phonetic matching, the Metaphone algorithm (Philips, 1990) was used (for details see Huber et al. (2006)).

[3]This approach can easily be generalized to multiple label chains.

4.3 Feature Generation

The annotation discussed above is the first step towards building a training corpus for a CRF-based approach. What remains to be done is to provide observations for each time step of the observed entity, i.e. for each token of a report; these are expected to give hints with regard to the annotation labels that are to be assigned to the time step. The observations, associated with one or more annotation labels, are usually called *features* in the machine learning literature. During CRF training, the parameters of these features are determined such that they indicate the significance of the observations for a certain label or label combination; this is the basis for later tagging of unseen reports.

We use the following features for each time step of the reports in C_{COR} and C_{RCG}:

- **Lexical features** covering the local context of ± 2 tokens (e.g., `patient@0`, `the@-1`, `is@1`)
- **Syntactic features** indicating the possible syntactic categories of the tokens (e.g., `NN@0`, `JJ@0`, `DT@-1` and `be+VBZ+aux@1`)
- **Bag-of-word (BOW) features** intend to capture the topic of a text segment in a wider context of ± 10 tokens, without encoding any order. Tokens are lemmatized and replaced by their UMLS *concept IDs*, if available, and weighed by `TF`. Thus, different words describing the same concept are considered equal.
- **Semantic type features** as above, but using UMLS *semantic types* instead of concept IDs provide a coarser level of description.
- **Relative position features**: The report is divided into eight parts corresponding to eight binary features; only the feature corresponding to the part of the current time step is set.

5 Structure Recognition with CRFs

Conditional random fields (Lafferty et al., 2001) are conditional models in the exponential family. They can be considered a generalization of multinomial logistic regression to output with non-trivial internal structure, such as sequences, trees or other graphical models. We loosely follow the general notation of Sutton and McCallum (2007) in our presentation.

Assuming an undirected graphical model G over an observed entity $\boldsymbol{x}$ and a set of discrete, interdependent random variables[4] $\boldsymbol{y}$, a conditional random field describes the conditional distribution:

$$p(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{x})} \prod_{c \in G} \boldsymbol{\phi}_c(\boldsymbol{y}_c, \boldsymbol{x}; \boldsymbol{\theta}_c) \qquad (1)$$

The normalization term $Z(\boldsymbol{x})$ sums over all possible joint outcomes of $\boldsymbol{y}$, i.e.,

$$Z(\boldsymbol{x}) = \sum_{\boldsymbol{y}'} p(\boldsymbol{y}'|\boldsymbol{x};\boldsymbol{\theta}) \qquad (2)$$

and ensures the probabilistic interpretation of $p(\boldsymbol{y}|\boldsymbol{x})$. The graphical model G describes interdependencies between the variables $\boldsymbol{y}$; we can then model $p(\boldsymbol{y}|\boldsymbol{x})$ via factors $\boldsymbol{\phi}_c(\cdot)$ that are defined over cliques $c \in G$. The factors $\boldsymbol{\phi}_c(\cdot)$ are computed from sufficient statistics $\{f_{ck}(\cdot)\}$ of the distribution (corresponding to the features mentioned in the previous section) and depend on possibly overlapping sets of parameters $\boldsymbol{\theta}_c \subseteq \boldsymbol{\theta}$ which together form the parameters $\boldsymbol{\theta}$ of the conditional distribution:

$$\boldsymbol{\phi}_c(\boldsymbol{y}_c, \boldsymbol{x}; \boldsymbol{\theta_c}) = \exp\left(\sum_{k=1}^{|\boldsymbol{\theta_c}|} \lambda_{ck} f_{ck}(\boldsymbol{x}, \boldsymbol{y}_c)\right) \qquad (3)$$

In practice, for efficiency reasons, independence assumptions have to be made about variables $y \in \boldsymbol{y}$, so G is restricted to small cliques (say, ($|c| \leq 3$). Thus, the sufficient statistics only depend on a limited number of variables $\boldsymbol{y_c} \subseteq \boldsymbol{y}$; they can, however, access the whole observed entity $\boldsymbol{x}$. This is in contrast to generative approaches which model a joint distribution $p(\boldsymbol{x}, \boldsymbol{y})$ and therefore have to extend the independence assumptions to elements $x \in \boldsymbol{x}$.

[4] In our case, the discrete outcomes of the random variables $\boldsymbol{y}$ correspond to the annotation labels described in the previous section.

The factor-specific parameters $\boldsymbol{\theta}_c$ of a CRF are typically tied for certain cliques, according to the problem structure (i.e., $\boldsymbol{\theta}_{c_1} = \boldsymbol{\theta}_{c_2}$ for two cliques c_1, c_2 with tied parameters). E.g., parameters are usually tied across time if G is a sequence. The factors can then be partitioned into a set of *clique templates* $\mathcal{C} = \{C_1, C_2, \ldots C_P\}$, where each clique template C_p is a set of factors with tied parameters $\boldsymbol{\theta}_p$ and corresponding sufficient statistics $\{f_{pk}(\cdot)\}$. The CRF can thus be rewritten as:

$$p(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \prod_{C_p \in \mathcal{C}} \prod_{\boldsymbol{\phi}_c \in C_p} \boldsymbol{\phi}_c(\boldsymbol{y}_c, \boldsymbol{x}; \boldsymbol{\theta}_p) \qquad (4)$$

Furthermore, in practice, the sufficient statistics $\{f_{pk}(\cdot)\}$ are computed from a subset $\boldsymbol{x}_c \subseteq \boldsymbol{x}$ that is relevant to a factor $\boldsymbol{\phi}_c(\cdot)$. In a sequence labelling task, tokens $x \in \boldsymbol{x}$ that are in temporal proximity to an output variable $y \in \boldsymbol{y}$ are typically most useful. Nevertheless, in our notation, we will let factors depend on the whole observed entity $\boldsymbol{x}$ to denote that all of $\boldsymbol{x}$ can be accessed if necessary.

For our structure recognition task, the graphical model G exhibits the structure shown in figure 3, i.e., there are multiple connected chains of variables with factors defined over single-node cliques and two-node cliques within and between chains; the parameters of factors are tied across time. This corresponds to the *factorial CRF* structure described in Sutton and McCallum (2005). Structure recognition using conditional random fields then involves two separate steps: *parameter estimation*, or training, is concerned with selecting the parameters of a CRF such that they fit the given training data. *Prediction*, or testing, determines the best label assignment for unknown examples.

5.1 Parameter estimation

Given IID training data $\mathcal{D} = \{\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}\}_{i=1}^{N}$, parameter estimation determines:

$$\boldsymbol{\theta}^* = \operatorname*{argmax}_{\boldsymbol{\theta}'} \left(\sum_{i}^{N} p(\boldsymbol{y}^{(i)}|\boldsymbol{x}^{(i)};\boldsymbol{\theta}')\right) \qquad (5)$$

i.e., those parameters that maximize the conditional probability of the CRF given the training data.

In the following, we will not explicitly sum over $_{i=1}^{N}$; as Sutton and McCallum (2007) note, the training instances $\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}$ can be considered disconnected components of a single undirected model G.

We thus assume G and its factors $\phi_c(\cdot)$ to extend over all training instances. Unfortunately, (5) cannot be solved analytically. Typically, one performs maximum likelihood estimation (MLE) by maximizing the conditional log-likelihood numerically:

$$\ell(\boldsymbol{\theta}) = \sum_{C_p \in \mathcal{C}} \sum_{\boldsymbol{\phi}_c \in C_p} \sum_{k=1}^{|\boldsymbol{\theta}_p|} \lambda_{pk} f_{pk}(\boldsymbol{x}, \boldsymbol{y}_c) - \log Z(\boldsymbol{x}) \tag{6}$$

Currently, limited-memory gradient-based methods such as LBFGS (Nocedal, 1980) are most commonly employed for that purpose[5]. These require the partial derivatives of (6), which are given by:

$$\frac{\partial \ell}{\partial \lambda_{pk}} = \sum_{\boldsymbol{\phi}_c \in C_p} f_{pk}(\boldsymbol{x}, \boldsymbol{y}_c) - \sum_{\boldsymbol{y}'_c} f_{pk}(\boldsymbol{x}, \boldsymbol{y}'_c) p(\boldsymbol{y}'_c | \boldsymbol{x}) \tag{7}$$

and expose the intuitive form of a difference between the expectation of a sufficient statistic according to the empiric distribution and the expectation according to the model distribution. The latter term requires marginal probabilities for each clique c, denoted by $p(\boldsymbol{y}'_c|\boldsymbol{x})$. Inference on the graphical model G (see sec 5.2) is needed to compute these.

Depending on the structure of G, inference can be very expensive. In order to speed up parameter estimation, which requires inference to be performed for every training example and for every iteration of the gradient-based method, alternatives to MLE have been proposed that do not require inference. We show here a factor-based variant of pseudolikelihood as proposed by Sanner et al. (2007):

$$\ell_p(\boldsymbol{\theta}) = \sum_{C_p \in \mathcal{C}} \sum_{\boldsymbol{\phi}_c \in C_p} \log p(\boldsymbol{y}_c | \boldsymbol{x}, MB(\boldsymbol{\phi}_c)) \tag{8}$$

where the factors are conditioned on the Markov blanket, denoted by MB[6]. The gradient of (8) can be computed similar to (7), except that the marginals $p_c(\boldsymbol{y}'_c|\boldsymbol{x})$ are also conditioned on the Markov blanket, i.e., $p_c(\boldsymbol{y}'_c|\boldsymbol{x}, MB(\boldsymbol{\phi}_c))$. Due to its dependence on the Markov blanket of factors, pseudolikelihood cannot be applied to prediction, but only to parameter estimation, where the "true" assignment of a blanket is known.

5.1.1 Regularization

We employ a Gaussian prior for training of CRFs in order to avoid overfitting. Hence, if $f(\boldsymbol{\theta})$ is the original objective function (e.g., log-likelihood or log-pseudolikelihood), we optimize a penalized version $f'(\boldsymbol{\theta})$ instead, such that:

$$f'(\boldsymbol{\theta}) = f(\boldsymbol{\theta}) - \sum_{k=1}^{|\boldsymbol{\theta}|} \frac{\lambda_k^2}{2\sigma^2} \quad \text{and} \quad \frac{\partial f'}{\partial \lambda_k} = \frac{\partial f}{\partial \lambda_k} - \frac{\lambda_k}{\sigma^2}.$$

The tuning parameter σ^2 determines the strength of the penalty; lower values lead to less overfitting. Gaussian priors are a common choice for parameter estimation of log-linear models (cf. Sutton and McCallum (2007)).

5.2 Inference

Inference on a graphical model G is needed to efficiently compute the normalization term $Z(\boldsymbol{x})$ and marginals $p_c(\boldsymbol{y}'_c|\boldsymbol{x})$ for MLE, cf. equation (6).

Using belief propagation (Yedidia et al., 2003), more precisely its sum-product variant, we can compute the beliefs for all cliques $c \in G$. In a tree-shaped graphical model G, these beliefs correspond exactly to the marginal probabilities $p_c(\boldsymbol{y}'_c|\boldsymbol{x})$. However, if the graph contains cycles, so-called loopy belief propagation must be performed. The message updates are then re-iterated according to some schedule until the messages converge. We use a TRP schedule as described by Wainwright et al. (2002). The resulting beliefs are then only approximations to the true marginals. Moreover, loopy belief propagation is not guaranteed to terminate in general – we investigate this phenomenon in section 6.5.

With regard to the normalization term $Z(\boldsymbol{x})$, as equation (2) shows, naive computation requires summing over all assignments of $\boldsymbol{y}$. This is too expensive to be practical. Fortunately, belief propagation produces an alternative factorization of $p(\boldsymbol{y}|\boldsymbol{x})$; i.e., the conditional distribution defining the CRF can be expressed in terms of the marginals gained during sum-product belief propagation. This representation does not require any additional normalization, so $Z(\boldsymbol{x})$ need not be computed.

[5] Recently, stochastic gradient descent methods such as Online LBFGS (Schraudolph et al., 2007) have been shown to perform competitively.

[6] Here, the Markov blanket of a factor ϕ_c denotes the set of variables occurring in factors that share variables with ϕ_c, non-inclusive of the variables of ϕ_c

5.3 Prediction

Once the parameters $\boldsymbol{\theta}$ have been estimated from training data, a CRF can be used to predict the labels of unknown examples. The goal is to find:

$$\boldsymbol{y}^* = \underset{\boldsymbol{y}'}{\operatorname{argmax}} \left(p(\boldsymbol{y}'|\boldsymbol{x}; \boldsymbol{\theta}) \right) \tag{9}$$

i.e., the assignment of $\boldsymbol{y}$ that maximizes the conditional probability of the CRF. Again, naive computation of (9) is intractable. However, the max-product variant of loopy belief propagation can be applied to approximately find the MAP assignment of $\boldsymbol{y}$ (max-product can be seen as a generalization of the well-known Viterbi algorithm to graphical models).

For structure recognition in medical reports, we employ a post-processing step after label prediction with the CRF model. As in Jancsary (2008), this step enforces the constraints of the `BIO` notation and applies some trivial non-local heuristics that guarantee a consistent global view of the resulting structure.

6 Experiments and Results

For evaluation, we generally performed 3-fold cross-validation for all performance measures. We created training data from the reports in C_{COR} so as to simulate a scenario under ideal conditions, i.e., perfect speech recognition and proper dictation of punctuation and headings, without hesitation or repetitions. In contrast, the data from C_{RCG} reflects real-life conditions, with a wide variety of speech recognition error rates and speakers frequently hesitating, repeating themselves and omitting punctuation and/or headings.

Depending on the experiment, two different subsets of the two corpora were considered:

- $C_{\{COR,RCG\}\text{-}ALL}$: All 2007 reports were used, resulting in 1338 training examples and 669 testing examples at each CV-iteration.
- $C_{\{COR,RCG\}\text{-}BEST}$: The corpus was restricted to those 1002 reports that yielded the lowest word error rate during alignment (see section 4.2). Each CV-iteration hence amounts to 668 training examples and 334 testing examples.

From the crossvalidation runs, a 95%-confidence interval for each measure was estimated as follows:

$$\bar{Y} \pm t_{(\alpha/2,N-1)} \frac{s}{\sqrt{N}} = \bar{Y} \pm t_{(0.025,2)} \frac{s}{\sqrt{3}} \tag{10}$$

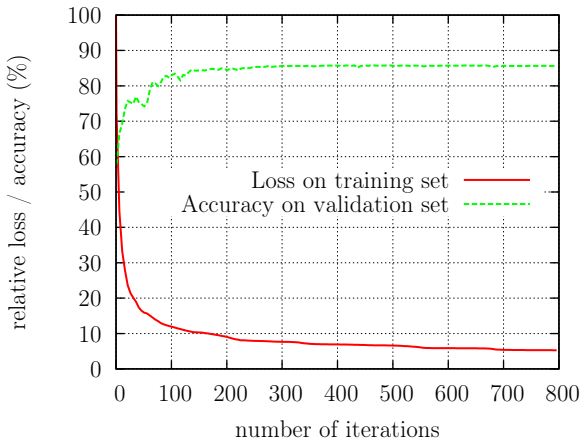

Fig. 5: Accuracy vs. loss function on $C_{RCG\text{-}ALL}$

where $\bar{Y}$ is the sample mean, s is the sample standard deviation, N is the sample size (3), α is the desired significance level (0.05) and $t_{(\alpha/2,N-1)}$ is the upper critical value of the t-distribution with $N-1$ degrees of freedom. The confidence intervals are indicated in the $\pm$ column of tables 1, 2 and 3.

For CRF training, we minimized the penalized, negative log-pseudolikelihood using LBFGS with $m = 3$. The variance of the Gaussian prior was set to $\sigma^2 = 1000$. All supported features were used for univariate factors, while the bivariate factors within chains and between chains were restricted to bias weights. For testing, loopy belief propagation with a TRP schedule was used in order to determine the maximum a posteriori (MAP) assignment. We use *VieCRF*, our own implementation of factorial CRFs, which is freely available at the author's homepage[7].

6.1 Analysis of training progress

In order to determine the number of required training iterations, an experiment was performed that compares the progress of the Accuracy measure on a validation set to the progress of the loss function on a training set. The data was randomly split into a training set (2/3 of the instances) and a validation set. Accuracy on the validation set was computed using the intermediate CRF parameters $\boldsymbol{\theta}_t$ every 5 iterations of LBFGS. The resulting plot (figure 5) demonstrates that the progress of the loss function corresponds well to that of the Accuracy measure,

[7] `http://www.ofai.at/~jeremy.jancsary/`

Estimated Accuracies		
	Acc.	±
Average	97.24%	0.33
Chain 0	99.64%	0.04
Chain 1	95.48%	0.55
Chain 2	96.61%	0.68
Joint	92.51%	0.97

(a) $C_{COR\text{-}ALL}$

Estimated Accuracies		
	Acc.	±
Average	86.36%	0.80
Chain 0	91.74%	0.16
Chain 1	85.90%	1.25
Chain 2	81.45%	2.14
Joint	69.19%	1.93

(b) $C_{RCG\text{-}ALL}$

Table 1: Accuracy on the full corpus

Estimated Accuracies		
	Acc.	±
Average	96.48%	0.82
Chain 0	99.55%	0.08
Chain 1	94.64%	0.23
Chain 2	95.25%	2.16
Joint	90.65%	2.15

(a) $C_{COR\text{-}BEST}$

Estimated Accuracies		
	Acc.	±
Average	87.73%	2.07
Chain 0	93.77%	0.68
Chain 1	87.59%	1.79
Chain 2	81.81%	3.79
Joint	70.91%	4.50

(b) $C_{RCG\text{-}BEST}$

Table 2: Accuracy on a high-quality subset

thus an "early stopping" approach might be tempting to cut down on training times. However, during earlier stages of training, the CRF parameters seem to be strongly biased towards high-frequency labels, so other measures such as macro-averaged F1 might suffer from early stopping. Hence, we decided to allow up to 800 iterations of LBFGS.

6.2 Accuracy of structure prediction

Table 1 shows estimated accuracies for $C_{COR\text{-}ALL}$ and $C_{RCG\text{-}ALL}$. Overall, high accuracy ($> 97\%$) can be achieved on $C_{COR\text{-}ALL}$, showing that the approach works very well under ideal conditions. Performance is still fair on the noisy data ($C_{RCG\text{-}ALL}$; Accuracy $> 86\%$). It should be noted that the labels are unequally distributed, especially in *chain 0* (there are very few BEGIN labels). Thus, the baseline is substantially high for this chain, and other measures may be better suited for evaluating segmentation quality (cf. section 6.4).

6.3 On the effect of noisy training data

Measuring the effect of the imprecise reference annotation of C_{RCG} is difficult without a corresponding, manually created golden standard. However, to get a feeling for the impact of the noise induced by speech recognition errors and sloppy dictation

Estimated WD		
	WD	±
Chain 0	0.007	0.000
Chain 1	0.050	0.007
Chain 2	0.015	0.001

(a) $C_{COR\text{-}ALL}$

Estimated WD		
	WD	±
Chain 0	0.193	0.008
Chain 1	0.149	0.005
Chain 2	0.118	0.013

(b) $C_{RCG\text{-}ALL}$

Table 3: Per-chain WindowDiff on the full corpus

on the quality of the semi-automatically generated annotation, we conducted an experiment with subsets $C_{COR\text{-}BEST}$ and $C_{RCG\text{-}BEST}$. The results are shown in table 2. Comparing these results to table 1, one can see that overall accuracy *decreased* for $C_{COR\text{-}BEST}$, whereas we see an *increase* for $C_{RCG\text{-}BEST}$. This effect can be attributed to two different phenomena:

- In $C_{COR\text{-}BEST}$, no quality gains in the annotation could be expected. The smaller number of training examples therefore results in lower accuracy.
- Fewer speech recognition errors and more consistent dictation in $C_{RCG\text{-}BEST}$ allow for better alignment and thus a better reference annotation. This increases the actual prediction performance and, furthermore, reduces the number of label predictions that are erroneously counted as a misprediction.

Thus, it is to be expected that manual correction of the automatically created annotation results in significant performance gains. Preliminary annotation experiments have shown that this is indeed the case.

6.4 Segmentation quality

Accuracy is not the best measure to assess segmentation quality, therefore we also conducted experiments using the WindowDiff measure as proposed by Pevzner and Hearst (2002). WindowDiff returns 0 in case of a perfect segmentation; 1 is the worst possible score. However, it only takes into account segment boundaries and disregards segment types. Table 3 shows the WindowDiff scores for $C_{COR\text{-}ALL}$ and $C_{RCG\text{-}ALL}$. Overall, the scores are quite good and are consistently below 0.2. Furthermore, $C_{RCG\text{-}ALL}$ scores do not suffer as badly from inaccurate reference annotation, since "near misses" are penalized less strongly.

	Converged (%)	Iterations (∅)
$C_{COR\text{-}ALL}$	0.999	15.4
$C_{RCG\text{-}ALL}$	0.911	66.5
$C_{COR\text{-}BEST}$	0.999	14.2
$C_{RCG\text{-}BEST}$	0.971	37.5

Table 4: Convergence behaviour of loopy BP

6.5 Convergence of loopy belief propagation

In section 5.2, we mentioned that loopy BP is not guaranteed to converge in a finite number of iterations. Since we optimize pseudolikelihood for parameter estimation, we are not affected by this limitation in the training phase. However, we use loopy BP with a TRP schedule during testing, so we must expect to encounter non-convergence for some examples. Theoretical results on this topic are discussed by Heskes (2004). We give here an empirical observation of convergence behaviour of loopy BP in our setting; the maximum number of iterations of the TRP schedule was restricted to 1,000. Table 4 shows the percentage of examples converging within this limit and the average number of iterations required by the converging examples, broken down by the different corpora. From these results, we conclude that there is a connection between the quality of the annotation and the convergence behaviour of loopy BP. In practice, even though loopy BP didn't converge for some examples, the solutions after 1,000 iterations where satisfactory.

7 Conclusion and Outlook

We have presented a framework which allows for identification of structure in report dictations, such as sentence boundaries, paragraphs, enumerations, (sub)sections, and various other structural elements; even if no explicit clues are dictated. Furthermore, meaningful types are automatically assigned to subsections and sections, allowing – for instance – to automatically assign headings, if none were dictated.

For the preparation of training data a mechanism has been presented that exploits the potential of parallel corpora for automatic annotation of data. Using manually edited formatted reports and the corresponding raw output of ASR, reference annotation can be generated that is suitable for learning to identify structure in ASR output.

For the structure recognition task, a CRF framework has been employed and multiple experiments have been performed, confirming the practicability of the approach presented here.

One result deserving further investigation is the effect of noisy annotation. We have shown that segmentation results improve when fewer errors are present in the automatically generated annotation. Thus, manual correction of the reference annotation will yield further improvements.

Finally, the framework presented in this paper opens up exciting possibilities for future work. In particular, we aim at automatically transforming report dictations into properly formatted and rephrased reports that conform to the requirements of the relevant domain. Such tasks are greatly facilitated by the explicit knowledge gained during structure recognition.

Acknowledgments

The work presented here has been carried out in the context of the Austrian KNet competence network COAST. We gratefully acknowledge funding by the Austrian Federal Ministry of Economics and Labour, and ZIT Zentrum fuer Innovation und Technologie, Vienna. The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry for Transport, Innovation, and Technology and by the Austrian Federal Ministry for Science and Research.

Furthermore, we would like to thank our anonymous reviewers for many insightful comments that helped us improve this paper.

References

ASTM International. 2002. ASTM E2184-02: Standard specification for healthcare document formats.

Freddy Choi. 2000. Advances in domain independent linear text segmentation. In *Proceedings of the first conference on North American chapter of the Association for Computation Linguistics*, pages 26–33.

C. Fellbaum. 1998. *WordNet: an electronic lexical database*. MIT Press, Cambridge, MA.

Marti A. Hearst. 1997. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):36–47.

Tom Heskes. 2004. On the uniqueness of loopy belief propagation fixed points. *Neural Comput.*, 16(11):2379–2413.

Martin Huber, Jeremy Jancsary, Alexandra Klein, Johannes Matiasek, and Harald Trost. 2006. Mismatch interpretation by semantics-driven alignment. In *Proceedings of KONVENS '06*.

Jeremy M. Jancsary. 2008. Recognizing structure in report transcripts. Master's thesis, Vienna University of Technology.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*.

S. Lamprier, T. Amghar, B. Levrat, and F. Saubion. 2008. Toward a more global and coherent segmentation of texts. *Applied Artificial Intelligence*, 23:208–234, March.

Vladimir I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710.

D. A. B. Lindberg, B. L. Humphreys, and A. T. McCray. 1993. The Unified Medical Language System. *Methods of Information in Medicine*, 32:281–291.

Evgeny Matsuov. 2003. Statistical methods for text segmentation and topic detection. Master's thesis, Rheinisch-Westfälische Technische Hochschule Aachen.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Flexible text segmentation with structured multilabel classification. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pages 987–994.

Jorge Nocedal. 1980. Updating Quasi-Newton matrices with limited storage. *Mathematics of Computation*, 35:773–782.

Lev Pevzner and Marti Hearst. 2002. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, 28(1), March.

Lawrence Philips. 1990. Hanging on the metaphone. *Computer Language*, 7(12).

L. R. Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77:257–286, February.

Scott Sanner, Thore Graepel, Ralf Herbrich, and Tom Minka. 2007. Learning CRFs with hierarchical features: An application to go. International Conference on Machine Learning (ICML) workshop.

Nicol N. Schraudolph, Jin Yu, and Simon Günter. 2007. A stochastic Quasi-Newton Method for online convex optimization. In *Proceedings of 11th International Conference on Artificial Intelligence and Statistics*.

Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47.

Charles Sutton and Andrew McCallum. 2005. Composition of Conditional Random Fields for transfer learning. In *Proceedings of Human Language Technologies / Empirical Methods in Natural Language Processing (HLT/EMNLP)*.

Charles Sutton and Andrew McCallum. 2007. An introduction to Conditional Random Fields for relational learning. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press.

Martin Wainwright, Tommi Jaakkola, and Alan S. Willsky. 2002. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *IEEE Transactions on Information Theory*, 49(5).

Ming Ye and Paul Viola. 2004. Learning to parse hierarchical lists and outlines using Conditional Random Fields. In *Proceedings of the Ninth International Workshop on Frontiers in Handwriting Recognition (IWFHR'04)*, pages 154–159. IEEE Computer Society.

Jonathan S. Yedidia, William T. Freeman, and Yair Weiss, 2003. *Understanding Belief Propagation and its Generalizations, Exploring Artificial Intelligence in the New Millennium*, chapter 8, pages 236–239. Science & Technology Books, January.