Sequence Models for Automatic Highlighting and Surface Information Extraction

Massih-Reza Amini, Hugo Zaragoza, Patrick Gallinari

LIP6, University of Paris 6,
case 169, 4 Place Jussieu, F-75252
Paris cedex 05, France.
{amini, zaragoza, gallinari}@poleia.lip6.fr

Abstract

With the increase of textual information available electronically, we assist to a great diversification of the demands on Information Retrieval (IR) and Information Extraction (IE) systems. In this paper we apply Machine Learning techniques of sequence analysis to the tasks of highlighting and labeling text with respect to an information extraction task. Specifically, dynamic probability models are used. Like IR systems, they use little semantics, are fully trainable and do not require any knowledge representation of the domain. Unlike IR approaches, documents are considered as a dynamic sequence of words. Furthermore, additional word information is naturally included in the representation. Models are evaluated on a sub-task of the MUC6 Scenario Template corpus. When morpho-syntactic word information is introduced into the representation, an increase in performances is observed.

1 Introduction

Information Retrieval (IR) deals with the selection, discrimination and representation of large bodies of documents. A document is considered as the basic data unit, and term occurrence frequencies are used to represent it. Information Extraction (IE) on the other hand, deals with the extraction of knowledge expressed in natural language within documents. It considers a word as the basic data unit, using morpho-syntactic as well as semantic information to represent each term. The nature of the representation of data and the models applicable to these two domains are very different, and so are their application goals; for this reason they have remained apart until recently. Nowadays, the demands for more versatile and precise IR systems, and more portable and domain-independent IE systems, have brought these two domains closer [2, 3, 11]. In our opinion, with the explosion of electronically available textual information, new and unforeseen demands are arising; many of these new tasks lie between the classic frameworks of IE and IR.

Machine Learning has a central role to play in the future of IE and IR. For the most part Machine Learning has been used, so far, to improve on existing frameworks, specially within the vector space model in IR and the concept-node approach in IE. In this paper we explore an alternative approach: the use of dynamic probability models for sequence analysis. Our present application of these models take into consideration the sequence of words of a document, not just their occurrence; for this reason the richness of individual word representations needs to be extremely reduced in order to keep the models computationally feasible. A surprising finding is that simple mappings, representing the set of terms on very low dimensional spaces, are sufficient for relatively complex tasks. An additional advantage of the sequence analysis formalism presented is that word sequences may be naturally extended to include different information sources, continuous or discrete in nature. To explore this approach, we experiment first with a basic word representation and then with the addition of morpho-syntactic information.

We deal in this paper with two example applications of sequence analysis with dynamic models: *text highlighting* and *surface information extraction*. In highlighting, we wish to select the most pertinent sequences of words within the text, given a specific task or information interest. In extraction, the information interest is subdivided in specific sub-interests, and text must be labeled with respect to these. We will deal with models that learn the task from a training corpus. We use Wall Street Journal articles from the MUC-6 corpus (and associated Scenario Templates) to test our models. We wish to highlight all the descriptions of *personnel change* events (job appointments, reassignments or destitutions) and, furthermore, extract the *name* and *position* of the person concerned.

Sequence Analysis is introduced in general in Section 2, the feature representation used is presented in Section 3. Models are presented in Section 4, followed by an evaluation of our models on the highlighting and extraction tasks in Section 5.

2 Sequence Analysis

In general terms, we wish to translate a known sequence of symbols (the input sequence) into an unknown sequence (the output sequence) representing some structural or abstract information contained in the known sequence. The input and output sequences may belong to different alphabets and obey different grammars. Grammars may be stochastic (non determinist), and the “alphabets” may in fact be continuous and represented in a vector space. Formally, we denote the input sequence as $w_{1,n} = w_1 w_2 \dots w_n$, $w_i \in \mathbf{W}$, and the output sequence as $t_{1,n} = t_1 t_2 \dots t_n$, $t_i \in \mathbf{T}$ where w_i and t_i denote the i th element of the input and output sequence respectively and $\mathbf{W}$ and $\mathbf{T}$ are the input and output spaces respectively. The elements of the input and output spaces are denoted as W_p and T_k respectively. In order to simplify the notation, we will write $P(t_i / w_j)$ to denote $P(t_i = T_k / w_j = W_p)$.

There exists a joint probability distribution over all the possible input and output sequences, which is unknown but can be inferred in some manner from a known set of labeled examples. What we wish to find is the most probable output sequence given an input sequence, i.e. we want to compute: $\arg \max_{t_{1,n}} P(t_{1,n} / w_{1,n})$. Different

models impose different assumptions to solve this estimation problem.

One may phrase the problem of Information Extraction into a problem of sequence analysis quite naturally if the number of *output symbols* or *classes*, i.e. the number of elements in $\mathbf{T}$ is known and fixed *a priori*, and there is assumed to be a one to one correspondence between words and output symbols. In such case, the input symbol sequence is the text itself (or some representation of it) and the output sequence is the sequence of labels that code the information classes contained in the text. Since we represent words as vectors, w_i is then the vector representation of the i th word of the sequence and t_i is its class index. $\mathbf{W}$ is the vector space of terms and $\mathbf{T}$ the set of information classes.

3 Feature Representation

A text will be considered as a sequence of terms, each of them being encoded in a distinct vector. The text will then be represented as the sequence of corresponding vectors. For this reason, we need a low-dimensional representation of terms. As proposed in [3], we use the discriminate U-measure to map terms into a one-dimensional continuous representation. Consider a phrase as the basic text unit, much like a document in standard IR; a relevant phrase is a phrase that contains relevant information and should be highlighted (as described in Section 5.1). For each term in the corpus, let n and n' denote respectively the number of relevant and irrelevant phrases in which the term appears. Let m and m' denote respectively the number of relevant and irrelevant passages in which the term does not appear. Then the U-measure of the term W_p is defined as [4]:

$$u_p = U(W_p) = \sqrt{N} \cdot \frac{nm' - n'm}{\sqrt{(n+n')(n+m)(n'+m')(m+m')}} \quad (1)$$

where N is the total number of passages ($N = n + n' + m + m'$). The U-measure is normalized over all marginal values ($n+n'$, $m+m'$, $n+m$, $n'+m'$). It has been used in IR for the selection of relevant terms [7]. Unlike the χ^2 -measure, the U-measure only rewards positive correlation. The measure was computed for each term on the training set (on stemmed words) and used for the test set. Terms with undefined U-measure and terms appearing just on the test set were not mapped.

The U-measure represents terms with respect to their frequency of occurrence in relevant and irrelevant phrases, and for this reason it represents (although very poorly) the *semantics* of terms with respect to the extraction task. This information is statistic and task-dependent, and furthermore it is relatively poor. Words with the same U-value may have different labels for a given surface extraction task and different U-values may represent terms that are in fact synonyms. In order to introduce a richer text representation, we have also

experimented with the use of morpho-syntactic information. We have augmented the previously described feature space of terms with morpho-syntactic tags. For this we have used a freely available probabilistic Part-Of-Speech (POS) tagger [13]. We retained the first 7 syntactic labels : Verb (V), Name (N), Proper Name (NP), Adjective (ADJ), Complement (CC), Determinant (DET), and words having another label were tagged as Others (O). Thus each term is represented as an 8 dimensional vector, the first component of which is the U-measure of the term, and the 7 others being 0 except the one corresponding to the tag of the term, which is set to 1.

All measurements proposed here are not equally informative, it is thus useful to use an automatic feature selection method on these variable sets. We have chosen here a method proposed in [8] to remove all non informative features. The relevance of a set of input variables is defined as the mutual information between these variables and the corresponding desired outputs. This dependence measure is well suited for measuring non linear dependencies. For two variables x and d , it is defined as: $MI(x, d) = \sum_{x,d} P(x, d) \log \frac{P(x, d)}{P(x)P(d)}$. Here x denotes an

input variable, i.e. an encoding of a term w , and d denotes an output symbol, i.e. a class associated to x (e.g. Irrelevant, Person or Position). Starting from an empty set, variables are added one at a time, the variable x_i selected at step i being the one which maximizes $MI(S_{v_{i-1}}, x_i, d)$ where $S_{v_{i-1}}$ is the set of $i-1$ already selected variables and d the desired output. Selection was stopped when the ratio $\rho = MI(S_{v_{i,p}}, d) / MI(S_{v_p}, d)$ raises above a fixed threshold (0.99 in our case). We performed variable selection for the two tasks and obtained the same set of variables in both cases. 5 variables were retained whose list in order of decreasing importance is: *U, NP, N, ADJ, V*.

4 Models

In section 2, our information extraction problems have been formulated in the framework of a sequence decoding problem. In the following we will consider different solutions for computing the most probable label sequence associated to a sequence of words. As stated in section 2, we wish to find $T_{1,n} = \arg \max_{t_{1,n}} P(t_{1,n} / w_{1,n})$.

$$T_{1,n} \text{ may be rewritten as } T_{1,n} = \arg \max_{t_{1,n}} P(t_{1,n}, w_{1,n}) \quad (1)$$

Following [9, 10], we will derive two main expressions from (1).

$$\text{First, (1) can be decomposed into: } T_{1,n} = \arg \max_{t_{1,n}} \prod_{i=1..n} P(t_i / t_{1,i-1}, w_{1,i}) P(w_i / t_{1,i-1} w_{1,i-1}) \quad (2)$$

$$\text{and under the assumptions: } P(w_i / t_{1,i-1} w_{1,i-1}) = P(w_i / w_{1,i-1}) \text{ and } P(t_i / t_{1,i-1}, w_{1,i}) = P(t_i / w_i) \quad (3)$$

$$\text{it reduces to : } T_{1,n} = \arg \max_{t_{1,n}} \prod_{i=1..n} P(t_i / w_i) \quad (4)$$

$$\text{Secondly, (1) can also be decomposed into: } T_{1,n} = \arg \max_{t_{1,n}} \prod_{i=1..n} P(t_i / t_{1,i-1}, w_{1,i-1}) P(w_i / t_{1,i}, w_{1,i-1})$$

$$\text{and under the assumptions: } P(t_i / t_{1,i-1}, w_{1,i-1}) = P(t_i / t_{i-1}) \text{ and } P(w_i / t_{1,i}, w_{1,i-1}) = P(w_i / t_i) \quad (5)$$

$$\text{it writes : } T_{1,n} = \arg \max_{t_{1,n}} \prod_{i=1..n} P(w_i / t_i) p(t_i / t_{i-1}) \quad (6)$$

Assumptions (3) or (5) are hardly verified for the problems we are interested in, however, they are often used to justify the derivation and use of stochastic sequence prediction models. Specifically, Hidden Markov Models and Neural Network models for sequence analysis implement equations (6) and (4) respectively, as described next.

4.1 Hidden Markov Models

In sequence analysis, a HMM is used to map the input (word) sequence into the output (class) sequence as follows. One considers that the HMM has emitted the input sequence, following a sequence of unknown (hidden) states. States map the alphabet of the output sequence, in our case the information classes, onto this input sequence. Therefore, determining a state sequence is equivalent to finding a labeling of the input sequence.

A first order HMM corresponds to hypothesis (5). Term emission probabilities $P(w_i = W_p / t_i = T_k)$ and state transition probabilities $P(t_i = T_k / t_{i-1} = T_k)$ are unknown, and need to be estimated. This is done by the EM algorithm with a training corpus of labeled data [9]. In the case of a continuous symbol space, probabilities are replaced by probability density functions, approximated by fitting a parametric model [6]. We used here gaussian mixture models, trained with the EM algorithm. The number of gaussians is fixed by cross-validation.

Not all state transitions are necessarily allowed. A *grammar* is used to constrain the allowed state transitions this is simply done by setting some transition probabilities to 0. Once we have trained each concept model independently, we construct a new HMM by concatenating the states as dictated by the grammar. The grammar is not learned, but rather, it is chosen by the model designer. The choice of an appropriate grammar is difficult, and critical for the performances of the system. Although it would be possible to use a completely unconstrained grammar (i.e. each word in a sequence may belong to any concept), as we introduce knowledge into the grammar we constraint the system, limiting the computational cost of parsing and reducing the number of sequences needed for training when the labeling is unknown.

As mentioned previously, the choice of grammar determines the sequence of concepts produced by the system. For highlighting we use the approach proposed in [1, 12], which consists in using a grammar of the type I-R-I, which finds the most probable single sub-sequence of relevant terms within the paragraph. While this does not correspond truly to the way relevant information is present in the MUC data, it provides a first approximation to the interest of our highlighting paradigm. Experiments with more sophisticated grammars did not led to improved performances. For information extraction words need to be spotted *and* labeled if they correspond to the fields POSITION and PERSON on a scenario template. Again the concepts are learned independently and they are then combined into a grammar model to segment sequences of text. For surface information extraction different grammars may be used depending on the available knowledge. Here we constrain the grammar to pursue the following state path: I - < Per - Pos - I >, where <> indicate one or more occurrences. Again, this choice of grammar does not match the MUC data, but it is the most common state sequence.

4.2 Multi-Layer Perceptrons

While eq. (6) required to estimate densities $P(w / t)$, implementing eq. (4) requires the computation of the posterior probabilities $P(t / w)$. Several models have been proposed in the pattern recognition literature for doing this. Over the last ten years, NNs [5] have established themselves as very efficient classification models. We will use here Multi-Layer Perceptrons (MLPs), which compute nonlinear transformations s.a.:

$$y_k(x) = f \left(\sum_{h=1}^H p_{kh} \cdot f \left(\sum_{i=1}^D p_{hi} \cdot w_i \right) \right),$$

where y_k is the k th component of the output vector, w is the input vector, p are the weights of the network (the model parameters), f is a one-dimensional squashing function, D is the dimension of the input space and H is the number of hidden units of the network. Parameters p are learned by gradient descent from a set of labeled data. The number of hidden units is usually set by cross validation.

If the desired output of a MLP is encoded as a binary vector whose components are all 0 but a 1 in position k when input w is from class k , it can be shown that a MLP approximates the posterior class probability $p(k/w)$. In this case, the output coordinate with higher value represents the most probable class of the object. This property is commonly used in pattern recognition as well as in NLP and IR applications.

Approximating $P(t / w)$ i.e. is assigning the most probable information class to each word, irrespectively of the word's context can be performed efficiently with MLPs. In this work, the drastic hypothesis (3) were slightly relaxed: $P(t_i / t_{1,i-1}, w_{1,i}) = P(t_i / w_i)$ in (3) was replaced by $P(t_i / t_{1,i-1}, w_{1,i}) = P(t_i / w_{i-k,i+k})$, which allows to take into consideration a local context in the current input sequence for predicting the tag associated to word w_i .

Estimates for this new posterior probability are easily obtained, provided the training set is representative of the task. Note that since MLPs interpolate to new data, these estimates can be computed even when the input context $w_{i-k,i+k}$ has never been seen in the training set. In the experiments, the size of the context was set up via cross validation. We also experimented with recurrent NNs which should, in principle, be able to learn automatically posterior estimates of $P(t_i / t_{1,i-1}, w_{1,i})$, but this did not lead to a performance improvement.

For the highlighting task, a one output-unit MLP is trained to discriminate between Irrelevant and Relevant segments of text. For extraction, a three outputs MLP is trained to discriminate between Irrelevant, Person and Position. The number of input units depends on the representation of terms and of the context size. In the experiments presented in Section 5 we used a window size of 5, and term representations of size 1 (U-measure) and 5 (U + relevant word tags).

5 Experiments

5.1 Data

In order to train and evaluate our models in the tasks of IE and highlighting, we need labeled data. More specifically, we need a collection of documents that have been analyzed (e.g. by human experts) and where the words relevant to each concept have been identified. We use the MUC6 Scenario Template (ST) corpus. This corpus consists of a set of Wall Street article journals and, for each document, a set of Scenario Templates. STs describe an instance of an event of interest (see Figure 1). In this case, STs describe instances of personnel changes (appointments, destitutions, etc.). STs are subdivided into fields that describes a particular aspect of the event (see Figure 1, top right). In the present work we concentrate on only two fields: the name and position on the person concerned. We designed an heuristic procedure to label automatically each word in the corpus with a class label. Since each document has an associated set of STs, and documents contain paragraphs delimiters, we compare each paragraph to each one of the document's ST. If the paragraph partially matches the contents of at least two ST fields, the paragraph is considered as relevant. Paragraphs are then subdivided into phrases (sequences of words separated by punctuation marks or conjunctions). If a phrase contains the text of an ST field, all the words in that phrase are given the class of the ST field. All the other words are labeled as being irrelevant. This heuristic leads to some labeling errors, insertions and replacements; furthermore, it makes the assumption that STs are filled from text coming from a single paragraph. We are interested in models capable of handling labeling errors typical of automatic labeling systems.

Figure 1 : A paragraph (top left) is compared with an Scenario Template (top right) to obtain word labels (bottom). In the Scenario Template box we have indicated in bold the two fields of interest: POST and PER_NAME. In the labeling box, PERSON labeled words are shown in bold and POSITION labeled words are shown in italics; IRRELEVANT words are shown in normal face and phrases endings are denoted by '/'.

This approach to constituting a labeled database to train and evaluate automatic IE models can be adopted to other situations besides MUC data. There are many common situations where we dispose of a database containing textual fields, and a corpus of documents from which the information proceeds or can be extracted. As an example, consider a database containing the name, position, courses taught and research description of the staff of a University, and the set of WWW University pages.

5.2 Evaluation

The evaluation of IE and highlighting is difficult and controversial. The full evaluation of our models is still in a preliminary stage. Given the definition of IE as a task of translating words into their classes, a straightforward performance measure that can be used is the percentage of words correctly classified by the system, for each class and overall, over a test set (a set of documents not used during training). This is of course a first approximation to the true performance of the system. The percentage of good classification (PGC) is defined as: c/N where c is the number of correctly classified words and N is the total number of words in the corpus. Since the examples of the irrelevant class are much more frequent than the examples of the rest of the classes, we will be interested as well in the PGC per class: c_k/N_k .

Presenting results in PGC hides the choice of a value of misclassification *risk* (implicitly set to unity, or the inverse class frequency, depending on the model). In IR, where the classes relevant and irrelevant are considered as inherently different and the risk of misclassifying a relevant document is much higher than the risk of misclassifying an irrelevant document, results are presented in the form of curves of precision recall. Precision is defined as c_k/M and recall as c_k/N_k , where M is the total number of words (correctly or incorrectly) judged relevant by the system. In this case the system does not discriminate between classes but rather gives a score on the relevancy of each word with respect to the class k . All documents are sorted by this relevancy, labeling the highest M as relevant and the rest as irrelevant. Precision and recall are computed for $M = 1, \dots, N$ and plotted against each other on a 11 point curve. We note that recall is equivalent to the PGC when discriminating between two classes; modifying the *risk* when computing the PGC is equivalent to modify M when computing the recall values.

5.3 Results

We have performed an evaluation of our models over the highlighting and the extraction tasks. Tests have been carried out to compare the two representation of texts described in section 3: U-measure and U-measure + morpho-syntactic tagging. The corpus was split into a training set and a test set. Altogether there were 105 and 100 paragraphs respectively in the training and the test set. About 8% of words are Person, 32% Position and 60% represent irrelevant terms. Table 1, shows the performances on the test set of the MLP for highlighting. The syntactic information contributes to a clear improvement of the discrimination.

	% correct classification	
	Relevant	Average
U-Measure	76.35	79.85
U-Measure + tagging	85.12	81.99

Table 1: Performance of the MLP model for the highlighting task. Average is over Relevant and Irrelevant classes.

	% correct classification		
	Position	Person	Average
U-Measure	63.41	50.28	67.98
U-Measure + tagging	64.37	52.94	70.16

Table 1 : Performance of the MLP model for the extraction task. Average is over Person, Position and Irrelevant classes.

For the extraction, performances are also improved by the addition of syntactic tags (Table 2). Although to a lesser extent. Many errors come from the Person label which is confused with the Position label. In Figure 2 we show the precision-recall curves obtained by the MLP module, where terms have been represented only by the U-measure (left) or by their U-measure plus the syntactic tags (right). For each representation, we give the recall-precision curves for the position and person classes (extraction task) as well as for the relevant class (highlighting task). The improvement when adding syntactic information is clearly seen.

Figure 2: Precision-Recall obtained by the MLP model, where terms are represented only by their U-measure (left) and their U-measure plus syntactic labels (right).

MLPs have been used to estimate posterior probabilities for implementing equation (4). We have implemented the alternative approach suggested by equation (6) using HMMs. Equation (6) requires the definition of a grammar over the HMMs states which define a set of allowed state sequences. This grammar determines the possible transitions between concepts. As described in section 4, for highlighting, we used I-R-I as grammar. The behavior of this slightly more complex model is similar to that of the MLP. Syntactic information provides a neat improvement over the relevant terms while the average performances decreases slightly.

% correct classification		
	Relevant	Average
U-Measure	65.59	64.09
U-Measure + tagging	85.15	62.73

Table 2 : Performance of the HMM model for the highlighting task. Average is over Position, Person and Irrelevant classes.

For extraction, we used the grammar described in Section 4.1. For this model, the mean performances are lower than that of the MLP, which is mainly due to the low performances on the person label. Syntactic information clearly helps to improve the performance on the Position class.

% correct classification			
	Position	Person	Average
U-Measure	55.86	18.27	57
U-Measure + tagging	87.49	12.29	57.37

Table 3 : Performance of the HMM model for the extraction task. Average is over Position, Person and Irrelevant classes.

It is clear that both models corresponding to equations (4) and (6) are over simplistic, the implementations we have

developed should be enhanced, in particular the definition of the grammars is crucial for the extraction task. However, the results of our experiments prove that these models, using a simple term mapping scheme, behave well for non trivial surface extraction tasks. Furthermore, we have shown that the addition of syntactic information in the term representation improves the performances of the systems.

6 Conclusion and perspectives

We have presented an application of Machine Learning techniques for sequence analysis to the problem of surface information extraction. Because of the trainable nature of these techniques, their relative simplicity and their robustness, they may be integrated with ease into more complex IR or Text Mining systems. We have given a detailed description of two different implementations and their evaluation, and shown that they are capable of relatively complex tasks. Furthermore, we have shown that morpho-syntactic information can be added in a straightforward manner and may yields a significant increase in the performances.

We are presently developing our work in two directions: the applications of machine learning models to new tasks in IR and IE, and the adaptation of these models, to go beyond our present simplistic use of dynamic models.

7 References

Journal Article

1. Mittendorf E. and Schauble P. Document Passage Retrieval Based on Hidden Markov Models. ACM SIGIR , 1994 ; pp. 318-327
2. Robertson A.M., Gaizuskas R. On the Marriage of Information Retrieval and Information Extraction. BCS-IRSG 1997; pp. 60-69
3. Zaragoza H. and Gallinari P. Coupled Hierarchical IR and Stochastic Models for Surface Information Extraction. BCG-IRSG 1998 ; pp.198-206

Book

4. Anderson E. The Statistical Analysis of Categorical Data. Springer, Berlin. 1992
5. Hertz J., Krogh A. and Palmer R.G. Introduction to the Theory of Neural Computation. Addison-Wesley Publishing Company, Redwood City, CA. 1991
6. Rabiner L., Juang B.H. (1993) Fundamentals of Speech Recognition. Prentice Hall, 1993

Paper in a Proceedings

7. Ballerini J.P., et al. SPIDER Retrieval System at TREC-5. In Proc. of the 5th Text Retrieval Conference, TREC-5, 1996
8. Bonnlander, Weigend A.S. Selecting Input Variables Using Mutual Information and Nonparametric Density Evaluation. In Proc. of ISANN'94, Taiwan, 1994, pp. 42-50
9. Charniak E. Statistical Language Learning. Bradford Book, The MIT Press, 1993

10. Charniak E., Hendrickson C., Jacobson N., Perkowitz M., (1996) *Equations for part of speech tagging*, Proceedings of the 11th National Conference on Artificial Intelligence, AAAI Press / MIT Press, 1996, pp. 784-789,
11. Grishman R., Sundheim B. Message Understanding Conference - 6: A Brief History. In Proc. Of the 16th International Conference on Computational Linguistics, 1996, vol. 1, pp. 466-471
12. Knaus D., Mittendorf E., Schauble P., Paraic S. Highlighting Relevant Passages for Users of the Interactive SPIDER Retrieval System. In Proc. Of the 4th Text Retrieval Conference, TREC-4, 1994
13. Schmid H. Probabilistic Part-of-Speech Tagging Using Descision Trees. In Proc. of International Conference on New Methods in Language Processing, 1994