Identifying Personal Health Information Using Support Vector Machines

Yikun Guo, Robert Gaizauskas, Ian Roberts, George Demetriou, Mark Hepple

Department of Computer Science, Sheffield University, Sheffield, UK

email: {initial.surname}@dcs.shef.ac.uk

Abstract

We explore the use of Support Vector Machines to recognize personal health information in medical discharge summaries. In addition to the basic token level features, we use entities recognized by an information extraction system designed for newswire text, plus a set of rules that incorporate entity-specific knowledge. The results on the unseen test dataset show that the SVM model can be easily adapted to a new domain with minimal work and achieve good performance (0.9869, the weighted F measure). The proposed new features also contribute to improving the accuracy of entity identification.

Introduction

Task 1 of the first Shared Task for Challenges in Natural Language Processing for Clinical Data is to automatically identify eight types of Personal Health Information (PHI) from medical discharge summaries. We view the task as a *Named Entity (NE) recognition* problem where the entities are the PHIs. Compared with the NE recognition problem in other domains, we observe the following difficulties: 1) medical discharge summaries are full of medical terminology, such as medical conditions, investigations undertaken, drugs used etc., which may cause more ambiguities between non-PHIs and PHIs than are found between entities in newswire text; 2) the summaries themselves are semi-structured, containing many fragmented sentences with misspelt names and foreign words, which increases the difficulties for an automatic NE recognition system; 3) in this de-identification challenge, real PHIs have been replaced with randomly generated surrogates, which can cause severe problems for dictionary-based methods, and as such can also challenge some of the more sophisticated machine learning approaches insofar as they use dictionary-based lookup as one source of information in identifying PHIs.

Conceptually, we view NE recognition as a classification task, in which for each token t_i in the sentence t_1, …, t_n, the classifier makes a binary decision y_i that t_i is or is not part of a NE. There is typically a strong dependency between the decision y_i for token t_i and the tokens $t_{i-j},\ldots,t_{i+j}$ that surround it, i.e. which form a local context. Our Support Vector Machine (SVM) classifier makes use of this local context in identifying the boundaries of NEs and assigning them to the pre-defined entity types.

SVM and NE Recognition

SVM [1] is a relatively new machine learning approach based on statistical learning theory. It is well-known for its good generalization performance and has been applied to many recognition problems. Recently, the SVM model has been applied to natural language processing tasks such as chunking [2] and text classification [3]. In particular, these two specific NLP systems are reported to have achieved higher accuracy than most state of the art systems (both learning and knowledge-based approaches). There are theoretical and empirical results that indicate that SVMs have the ability to generalize in a high dimensional feature space without over-fitting the training data [3]. Using a SVM is a natural choice for NE recognition because the attribute vectors are very high dimensional and sparse.

To represent NEs, we adopt the BIO representation scheme, originally proposed by [4], which allows *multi-token* NEs to be captured by tags marked on *individual* tokens. The following tags are used:

I	the current word is inside an entity chunk
O	the current word is outside an entity chunk
B	the current word is the beginning of a chunk which immediately follows another chunk.

For example, in the sequence below (whose elements take the form "token/tag"), tokens t_2-t_3 form an entity, as indicated by a continuous sequence of I-tags (bounded by O-tags). The continuous sequence of 'in chunk' tags across t_5-t_8, however, contains a B-tag, signalling a boundary, and so the interpretation is that t_5 forms one chunk and t_6-t_8 another.

t_1/O t_2/I t_3/I t_4/O t_5/I t_6/B t_7/I t_8/I t_9/O

Standard SVM learning constructs only binary classifiers. To build a multi-class NE recogniser, we provide a separate classifier for each PHI class. Using the BIO scheme, each PHI class requires three labels to be assigned (i.e. B/I/O), and this in turn is achieved using a one-vs-rest method, which uses three binary SVMs, which assign a different one of the three labels to tokens, and whose outputs are combined by a post-processing procedure.

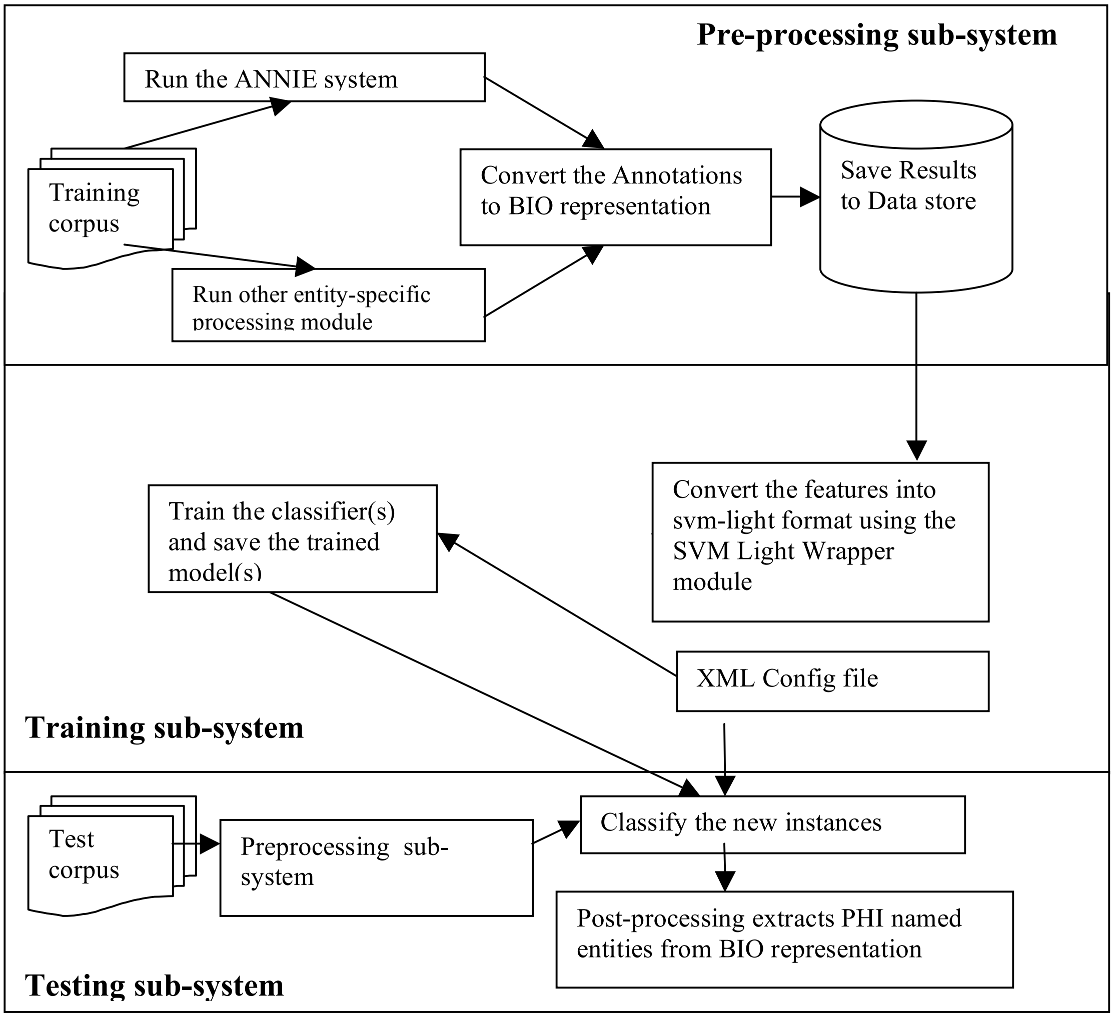
Pre-processing sub-system
Run the ANNIE system
Training corpus
Convert the Annotations to BIO representation
Save Results to Data store
Run other entity-specific processing module
Train the classifier(s) and save the trained model(s)
Convert the features into svm-light format using the SVM Light Wrapper module
XML Config file
Training sub-system
Test corpus
Preprocessing sub-system
Classify the new instances
Post-processing extracts PHI named entities from BIO representation
Testing sub-system

Figure 1: The architecture of the de-identification system

System Description

Figure 1 depicts the core functions of our de-identification system. We use the GATE system [5] to create and manage lexical and other meta-information about the text, known as annotations, and to convert the annotations to features for use in SVM classification. For the latter task, we use the SVM Light system [6], which has been 'wrapped' for use as a GATE module.

- The Preprocessing sub-system

In the preprocessing stage, we use ANNIE (A Nearly-New IE system) [5], an information extraction (IE) system that is distributed freely with GATE, to tokenize the input texts, split sentences, look up tokens in its gazetteer lists, produce part-of-speech tags and finally to identify entities based on JAPE grammar rules. The predefined entity types ANNIE recognizes, including person name, date, address, location, job title etc. overlap with the entity types defined for the challenge, but are not defined in precisely the same way, and so applying ANNIE directly will not yield good results. For example, in the challenge, date entities include only month and day, whilst those found by ANNIE may include also year and time. Another reason for not using ANNIE directly is that it depends heavily on dictionary-based lookup, and so the challenge's strategy of using randomly generated surrogates for PHIs will make it perform rather worse than it will on real data.

After running ANNIE, using the default parameters, we use customized JAPE rules and a customized dictionary to add entity-specific annotations, as will be discussed in detail in the next section. The resulting annotations for each token, including its PHI class, are then converted to the BIO representation and stored for use in SVM training.

- The training subsystem:

After pre-processing the corpus, the annotations are loaded from the data store and converted to the appropriate input format for SVM Light, which consists of real-valued feature-value pairs. This conversion is handled by the SVM Light Wrapper module that comes with GATE, in a way that is determined by an xml config file, which specifies the annotations to be converted, the size of the surrounding context and the weight of each annotation. The wrapper then calls SVM Light to train the SVM model.

- The testing subsystem:

In the application stage, the test corpus is first run through the pre-processing subsystem and the annotations are converted into features, just as with the training corpus. The SVM classifier then classifies the unseen instances, based on the model created in training, as well as the same xml config file. A post-processing module is then applied to determine the entities indicated by the BIO outputs of the different PHI class classifiers.

Feature selection

Because of time limitations, we submitted only two runs. One run used only basic token level features, such as word root, part of speech, etc. The other run used further information, in addition to the token level features. The following section first describes the basic token level features, and then explains, for each PHI type, the additional features used.

Token level features for PHI entity recognition

The input to SVM Light consists of sparse vectors of reals. Each feature of input data instances is mapped to some position in the vector and its possible values transformed to numeric values (e.g. integers). The basic token level features are as follows:

- root: the root string of a token;
- orth: the orthographic character of a token, with four possible values: “upperInitial”, “allCaps”, “lowercase” and “mixedCaps”.
- affix: affix information obtained by applying a morphological analyzer module;
- length: the character count of a token;
- part-of-speech: tag assigned by the POS tagger provided with GATE, from Hepple [7]. This tagger was trained on newswire text. It has not been re-trained for use with patient record summaries;
- kind: a feature that can take one of the three values “word”, “number” or “punctuation”, depending on the type of the token.

Additional features used

In addition to the basic token level features, we employ a different set of additional features for each entity type, which was determined empirically.

- Date: the classifier for dates uses two information sources additional to the basic token level features. Firstly, ANNIE date entities are used as a feature, although (as stated earlier) their definition differs from that of dates in the challenge. Secondly, a set of JAPE rules were added to recognize possible date entities in the text, which are in a suitable date format (e.g. “number/number” or “number-number”), and have month and day values in the valid range (e.g. such as “10/15, but not “55/100”). The two kinds of date entity are transformed to the BIO representation to provide input to the SVM.
- ID: ID entities usually are a series of numbers. We used only the token-level features and experimented with the size of the context.
- Phone: there is no corresponding ANNIE type for PHONE. A set of JAPE rules are included to recognize possible PHONE numbers in forms like “number(3)-number(3)-number(4)”, “number(3)-number(4)”, “number(3) number(3) number(4)” and “(number(3)) number(3)-number(4)” etc., where number(i) means a number that is i characters in length. Recognized Phone entities are then transformed into BIO representation.
- Doctor: in addition to token level features, we tried using ANNIE Person entities. The ANNIE Person name recognizer uses lookup in a name dictionary and JAPE rules to identify names in the texts. We observed that this recognizer did not perform well on the challenge data, probably because it was developed mainly for the newswire domain, which is somewhat different in style to medical discharge summaries. Furthermore, the ANNIE name recognizer does not distinguish doctor names from patient names, and so it can only act as a weak indicator of possible doctor name occurrences in the texts. ANNIE Title entities were also added, to reward names that have medical titles, such as “Dr.” or “M.D.”, occurring within the context. Also, lists were included of variations of medical titles, e.g. such as “dr” and “dr.” for “Dr.”, and to cover the special tokens “TR :” and “DD :”, which commonly appear as the start and end of doctor names. Finally, we also collected a sample of negative cue terms, such as “UNKNOWN PHYSICIAN”, which may appear at the position where a doctor name might be expected. These “negative” cue words were transformed into the BIO representation.
- Patient: surprisingly, patient name is much easier for the SVM to identify. Using only token level features and increasing the context window to

size -6/+6 (i.e. 6 tokens to the left and right of the current token) is enough to produce a fairly satisfactory level of performance. This may be because patient names tend to appear in rather fixed positions, e.g. following the string "**** DISCHARGE ORDERS ****". Our experiments showed that adding the ANNIE Person and Title types made performance slightly worse.

- Hospital: ANNIE organization entities were included as a feature, although this type covers not only hospital names, but also many other kinds of organization, such as company, university, etc. For the hospital type defined in the challenge, we observed occurrences of pre-modifier cue words, such as "CARE SITE" and "POSTPARTUM CARE SITE", post-modifier cue words, such as "Service :" and cue words appearing in the hospital name, such as "Medical Center" and "Health Center". Lists were created to store these three kinds of cue words, so they could be identified in texts and used as a feature.
- Location: Firstly, ANNIE location entities were included as a feature. Then it was observed that locations often occur after hospital names in texts, to indicate the latter's geographic location, and so features used for identifying hospital names were also added. Next, we made use of the ANNIE Lookup annotation, converting some location-related lookup annotations into the BIO representation for input to SVM learning. This included, for example, US city names and state names and street cue words, such as "St." and "Ave." etc. Lastly, the ANNIE Unknown annotation, which mainly consists of proper names that cannot be classified into any ANNIE types, is also added as a feature to improve recall, since location entities are sometimes recognized as Unknown by ANNIE.
- Age: this entity type is a special one in the challenge not only because it is rare in the training corpus, but also because, according to HIPAA, only ages greater than 90 are regarded as PHI, so as to require de-identification. In addition to the token level features, JAPE rules were included to identify candidate age expressions, constrained both by format, e.g. as in "numbery" or "number-year-old", and requiring the numeric value to be in a relevant range, i.e. 90 to 140. Unfortunately, because we had a very tight timeline (we started just one week before the deadline), we did not have time to submit the Age identification results for both runs.

Evaluation

Table 1 summarizes the official performance results on the unseen test dataset. For these experiments, SVM Light's parameter d was set to value 3. Due to time constraints, we only worked on a subset of the training data in the development stage to accelerate the feature selection procedure and to determine the optimum context window size, and we used only a subset of the training corpus to train the SVM model before applying it to the test dataset.

The last column of the table distinguishes experiments as being either `run 1', where only token level features were used, or as `run 2', where additional features were also included (as listed in the Feature column). Due to time constraints, we did not submit run 2 results for the ID entity, nor run 1 results for the Location entity, and Age results were submitted for neither. These four missing sets of results affect the overall system performance scores.

The unweighted system F-measure for run 1 is 0.686 and 0.672 for run 2, while the weighted F measure for both runs are 0.987 and 0.982 respectively.

For every type, we first used the basic token level features to start, and then varied the width of the context window to identify the best window size, and after that next varied the size of the data used for training, again to determine the optimum amount. The additional features to be used were then selected by error analysis or by exploring the existing ANNIE annotations. The results in the table show that there is usually an obvious performance improvement (shown in round brackets), when ANNIE existing entities, customized JAPE rules and other entity-specific information are used.

Conclusion

In this de-identification challenge task, we have experimented with training SVM classifiers for PHI entity types by combining three sources of information, i.e. basic token level information, the existing ANNIE results and newly created JAPE rules as well as augmented lists. We did not change the existing ANNIE system at all, which was designed mainly for newswire texts; instead we fed the relevant annotations directly to the SVM model and allowed the model to adapt itself to the new domain automatically. Our experiments show that the SVM aproach is very good at learning from imperfect data and can achieve good performance whilst using only a subset of the training data.

In the future, we will try to train the model using the entire corpus to see if the performance improves further. Initial explorations suggest that our system sometimes performs less well when trained with all the available training data rather than some portion of it, hinting at a problem of over-fitting. We will explore strategies for making use of the full training data and maximizing the performance whilst avoiding over-fitting.

Entity Type	Feature	Training Set used (Docs)	Context windows size	Precision	Recall	F Measure	Note
Date	Token	200	-5, +5	95	84	89	Run1
	Token + ANNIE Date + JAPE Date Rules	200	-5, +5	98	94	96 (+7)	Run2
ID	Token	200	-4,+4	72	95	82	Run1
		300	-4,+4	0	0	0	N/A
Phone	Token	200	-4,+4	73	84	78	Run1
	Token + JAPE Phone Rules	200	-4,+4	97	81	88 (+10)	Run2
Doctor	Token	200	-4,+4	95	88	92	Run1
	Token + ANNIE Person + ANNIE Title + cue words + negative cue words	200	-3, +3	97	94	96 (+4)	Run2
Patient	Token	200	-6, +6	96	90	93	Run1
	Token + ANNIE Person + ANNIE Title	200	-6, +6	95	92	93(+0)	Run2
Hospital	Token	200	-5, +5	85	82	83	Run1
	Token + ANNIE Organization + cue words	200	-4, +4	83	87	85 (+2)	Run2
Location	Token	200	-3, +3	0	0	0	N/A
	Token + ANNIE Location + Hospital features + ANNIE Lookup + ANNIE Unknown	200	-3, +3	60	38	47	Run2
Age	Token	200	-2, +2	0	0	0	N/A
	Token + JAPE Age rule	200	-3, +3	0	0	0	N/A

Table 1: Official system performance of the SVM approach applied to the unseen test data

Acknowledgements

This activity is part of the CLEF project and is funded by the UK Medical Research Council grantID: 60086.

References

1. Vladimir N. Vapnik. The Nature of Statistical Learning Theory. Springer-Verlag, New York; 1995.
2. Taku Kudo and Yuji Matsumoto. Chunking with Support Vector Machines. In Proceedings of the NAACL 2001;192-199; 2001.
3. Thorsten Joachims. Learning to classify text using support vector machines – methods, theory, and algorithms. Kluwer; 2002.
4. Lance A. Ramshaw and Mitchell P. Marcus. Text chunking using transformationbased learning. In Proceedings of the 3rd Workshop on Very Large Corpus; 88-94; 1995.
5. H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02). Philadelphia; July 2002.
6. T. Joachims, Making Large-Scale SVM Learning Practical. In Advances in Kernel Methods - Support Vector Learning, B. Scholkopf, C. Burges and A. Smola (ed.), MIT-Press, 1999.
7. M. Hepple. Independence and commitment: Assumptions for rapid training and execution of rule-based POS taggers. In Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL-2000), Hong Kong; October 2000.