ISSN:2249-5789

Examining a Pipelined Approach for Information Extraction with respect to machine learning

Mehnaz Khan
Research Scholar
Department of Computer Science
University of Kashmir

Dr. S.M.K. Quadri
Director
Department of Computer Science
University of Kashmir

Abstract

Pipelining is a process in which a complex task is divided into many stages that are solved sequentially. A pipeline is composed of a number of elements (processes, threads, co routines, etc.), arranged in such a way so that the output of each element is fed as input to the next in the sequence. Many machine learning problems are also solved using a pipeline model. Pipelining plays a very important role in applying the machine learning solutions efficiently to various natural language processing problems. The use of pipelining results in the better performance of these systems. However, these systems usually result in considerable computational complexity. For this reason researchers were motivated for using active learning for these systems. Reason of using active learning is that these algorithms perform better than the traditional learning algorithms keeping the training data same. In this paper we discuss an active learning strategy for pipelining of an important natural language processing task i.e. information extraction.

1. Introduction

A number of natural language processing applications use machine learning algorithms. These applications include parsing, semantic role labelling, information extraction, etc. Using a machine learning algorithm for one natural language processing task often requires the output from another task. Thus we can say these tasks are dependent on one another and therefore must be pipelined together. Therefore, a pipeline organization is used to model such situations. The benefit of using such an organization includes its ease of implementation and the main drawback is accumulation of errors between the stages of the pipeline that considerably affects the value of the results [4]. Pipelining has been used for a number of natural language applications e.g. bottom-up dependency parsing [11], semantic role labelling [8]. A bidirectional integration of pipeline models has been developed as a solution to the problem of error accumulation in traditional pipelines [10]. In this paper we show pipelining of information extraction. Although work has been done earlier in this regard which show pipelining of entity detection and relation extraction stages of information extraction, however, not much has been done with regard to part-of-speech tagging. One of the important contributions with regard to pipelining of information extraction includes that of Roth and Small (2008) who have given a method in which they combine separate learning strategies from a number of pipelined stages into a single strategy [2]. Here we theoretically discuss about including part-of-speech tagging stage of information extraction into the pipeline. We first give a general overview of the information extraction process in Section 2 along with an example to show how the process will work. In Section 3 we discuss about some of the work done in this field earlier and the problems faced by using supervised learning for information extraction. Those problems are the main reasons for preferring active learning approach. In the later sections we discuss machine learning and pipelining and also the reason why we suggest incorporating part-of-speech tagging in the pipelining process.

2. Simple Architecture of Information Extraction

Information extraction (IE) can be defined as a process which involves automatic extraction of structured information such as entities, relationships between entities, and attributes describing entities from unstructured and/or semi-structured machine-readable documents [5]. It can also be defined as a process of retrieving relevant information from documents. Applications of IE include news tracking [12], customer care [9], data cleaning [1], and classified ads [13]. Figure 1 shows a simple architecture of information extraction system [7]. The overall process of information extraction is composed of a number of subtasks such as segmentation, tokenization, part of speech tagging, named entity recognition, relation extraction, terminology extraction, opinion extraction, etc.

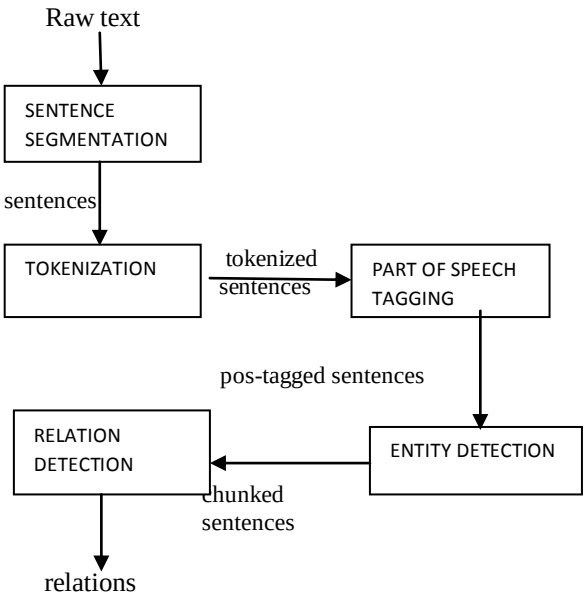

Figure 1: Simple Architecture of Information Extraction System

These subtasks of information extraction can be implemented using a number of different algorithms e.g. list-based algorithms for extracting person names or locations [18], rule-based algorithms for extracting phone numbers or mail addresses, and advanced machine learning and statistical approaches for extracting more complex concepts.

Sentence segmentation is the process of breaking the text into component sentences. Tokenization breaks the text into meaningful elements such as words, symbols. This is followed by part-of-speech tagging as shown in Figure 1 which labels these tokens with their POS categories. An example of applying these steps to a piece of text is shown below:

Jake works in Calgary, Alberta with his brother Micheal.

Jake	works	in	Calgary	Alberta
NP	VB	P	NP	NP

With	his	brother	Micheal
P	DET	NP	NP

Figure 2. Tokenization and Labelling

This is followed by entity detection. It is the process of identifying the entities having relations between one another, e.g. considering the above sentence, entities are detected as follows:

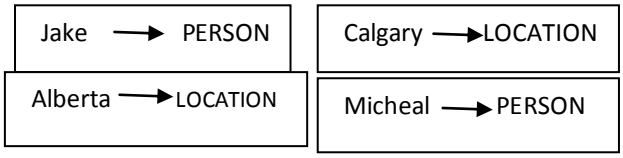

Figure 3: Entity Detection

Finally, after entities have been identified, the relations that exist between them are extracted in the relation detection step as follows:

{Jake, Calgary} → works_in
{Jake, Micheal} → brother_of
{Calgary, Alberta} → located_in
{Jake, Alberta} → works_in

Relation Detection

3. Related Work

Using pipelining in modelling the process of information extraction has resulted in an increase in efficiency. A lot of work has been done in this regard. Roth and Small have proposed a model that has demonstrated a significant reduction in supervised data requirements [2]. Efficient information extraction pipelines have been developed that have resulted in the efficiency gains of up to one order of magnitude [15]. A pipeline-based system has been developed for automated annotation of Surgical Pathology Reports [6]. There has been a lot of research in the field of information extraction using supervised machine learning. A number of supervised approaches have been proposed for the task of relation extraction which consists of some feature based methods [27, 14] and kernel methods [19, 3]. However, supervised methods have a number of disadvantages. First of all, we cannot extend these methods to define new relations between the entities due to lack of new labeled data as supervised methods have a predefined set of labeled data. Same problem occurs if we wish to extend the entity relations to higher order. Also for large input data these methods are computationally infeasible [16]. One of the main disadvantages of using supervised methods is the high cost associated with them as they require large amounts of annotated data. Active learning [20] provides a way to reduce these labeled data requirements. These algorithms are capable of collecting new labeled examples for annotation by making queries to the expert. The main advantage of using pipelining is that when the pipelining process starts the examples that are selected first are those that are needed at the beginning phases of pipeline followed by those that are needed later.

4. Pipelining and Machine Learning

In the supervised machine learning problem a function maps the inputs to the desired outputs by determining which of a set of classes a new input belongs to. This is determined on the basis of the training data which contains the instances whose class is known e.g. classification problem. The mapping function can be represented by *f*. *h* denotes the hypothesis about the function to be learned. Inputs are represented as $X = (x_1, x_2, \ldots, x_n)$ and outputs as $Y=(y_1, y_2, \ldots, y_n)$ [17]. Therefore, hypothesis or the prediction function can be written as

$$h : X \longrightarrow Y$$

h is the function of vector-valued input and is selected on the basis of training set of m input vector examples i.e.

$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) \longrightarrow$ [h] $\longrightarrow h(\mathbf{X})$

Training set = { $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_m$}

Therefore, the predicted value can be given as

$$y = h(x) = \text{argmax}_{y' \epsilon Y} f(x, y')$$

In case of pipelining, we have different stages. Let there be *N* stages. Therefore, each stage n depends on the previous *(n-1)* stages i.e.

$$x, y^{(0)}, \ldots, y^{(n-1)} \longrightarrow x^{(n)}$$

Therefore, in case of pipelining the predicted value can be written as

$$y = h(x) = [\text{argmax } f^{(n)}(x^{(n)}, y')]$$

where n = 1,…, N.

As discussed earlier in this paper, active learning algorithms reduce the number of labeled examples needed to learn any concept by collecting new unlabelled examples for annotation [21]. The examples are selected from the unlabelled data source U and are then labeled and added to the set of labeled data L [20]. Figure 4 shows the process of active learning [25]. The examples are selected by making queries to the expert. Query strategies that have been used earlier are uncertainty sampling [23] and query by committee [26]. In both these strategies the point is to evaluate the informativeness of the unlabeled examples.

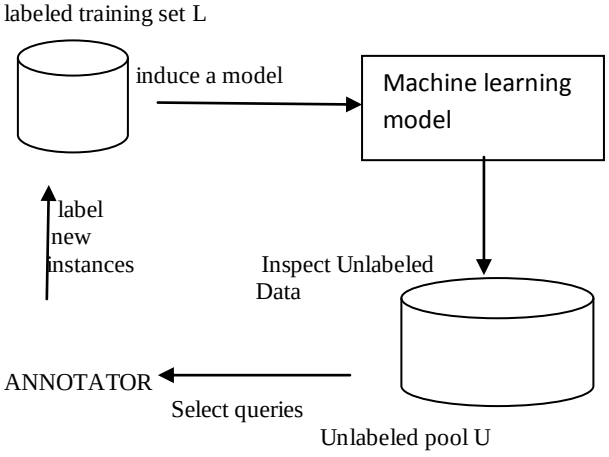

Figure 4: Pool Based Active Learning

The most informative instance or best query is represented as x^*_A, where A represents the query selection method used [20]. In uncertainty sampling, the algorithm selects that example about which it is least confident. In that case,

$$x^*_{LC} = \text{argmax } 1 - P_\theta (y \mid x) \quad [24]$$

In case of margin sampling,

$$x^*_M = \text{argmin } P_\theta(y_1 \mid x) - P_\theta(y_2 \mid x) \quad (1)$$

where y1 and y2 are first and second most probable class labels [22].

Another uncertainty sampling strategy that uses entropy as uncertainty measure,

$$x^*_H = \text{argmax } - \Sigma_i P_\theta(y_i \mid x) \log P_\theta(y_i \mid x) \quad (2)$$

where y_i represents all the class labels [20]

Scoring functions are also used for selecting the examples to be labeled or annotated. Scoring functions are used for mapping an abstract concept to a numeric value. Here, the idea is to calculate the score values for each instance to be labeled and the one with the minimum value is selected [2] i.e.

$$x^* = \text{argmin } q(x)$$

where x is selected from the unlabeled data U.

Therefore, for each stage n of the pipeline, there is a separate querying function i.e. $q^{(n)}$, and after combining all these functions we get,

$$x^* = \text{argmin} \Sigma q^{(n)}(x)$$

where n = 1,.., N and x belongs to U and N is the total number of stages of a pipeline. The pipelining process using active learning consists of the following steps:

1. As discussed earlier, each stage n of the pipeline has its own querying function $q^{(n)}$ and learner $l^{(n)}$. First of all, for each stage n, the hypothesis function as well as the querying function is estimated.
2. The unlabelled examples or instances are then selected by the learner from unlabeled data U and after labeling are added to labeled data L for each stage n of the pipeline.
3. As L changes after annotation of new instances, hypothesis is modified accordingly for each stage n.
4. The process is repeated until the final hypothesis is obtained after all the N stages of pipeline have been completed.

5. Stages of Information Extraction used in Pipelining

Pipelining has been applied to information extraction earlier where the focus has been on entity detection and relation extraction. But as far as part-of-speech tagging is involved, not much has been done towards

including it in the pipelining process of information extraction. Each stage of a pipeline is dependent on the earlier stages. In pipelining of information extraction, entity detection and relation detection highly depend on part-of-speech tagging. As discussed earlier, part-of-speech tagging labels each word or phrase of a sentence with its POS category. It helps in recognizing different usages of the same word and assigns a proper tag e.g. in the sentences below the word 'protest' has different usages:

The protest is going on. (Noun)
They protest against the innocent killings. (Verb)

Including part-of-speech tagging in the pipeline using active learning will result in the performance gain as the machine learning methods used for part-of-speech tagging have resulted in more than 95% accuracy. Moreover, in any natural language there are a number of words that are part-of-speech ambiguous (about more than 40%) and in such cases automatic POS tagging makes errors and hence require the use of machine learning techniques for tagging.

As discussed earlier, part-of-speech tagging labels each word or phrase of a sentence with its POS category, entity detection identifies the entities having relationships between one another in the sentence and relation detection extracts those relationships. Hence, in all these processes sentences are selected and annotated for all stages of the pipeline.

5.1. Including POS Tagging in Pipelining

In this section we theoretically show how active learning would be applied to POS tagging. As discussed earlier, first the informativeness of the unlabeled instances, sentences in our example, would be evaluated. Sentences would be selected from the unlabeled data and annotated/labeled by the annotator i.e. each word in the sentence would be tagged by its appropriate POS category. The annotated sentences will then be added to the labeled data. In Query By Uncertainty (QBU) approach, the informativeness of the unlabeled instances/examples is determined by evaluating the entropy- a measure of uncertainty associated with a random variable. In our example, these unlabeled instances are sentences. Therefore, we have to evaluate the entropy of sequence of words w_i in a sentence of length n, i.e.

$$H(w_1,w_2,\ldots,w_n) = -\Sigma\ p(w_1,w_2,..,w_n)\ \log p(w_1,w_2,\ldots,w_n)$$

From equation (2) we get,

$$x^*_H = -\Sigma\ p(y_i \mid x) \log p(y_i \mid x)$$

for each word w_i of the sentence, pos_i represents the part-of-speech tag for that word. Thus, the querying function for the part-of-speech tagging stage will be given as

$$q_{pos} = -\Sigma\ p(pos_i \mid w_i, y_i, pos_{i-1}, pos_{i-2}) \log p(pos_i \mid w_i, y_i, pos_{i-1}, pos_{i-2})$$

where i = 1 to n and posi-1 and pos_{i-2} represent the tags of previous two words.

5.2. Active learning for Entity and Relation Detection

For this stage too QBU approach will be used which selects those unlabeled examples/instances about which the learner is least confident. According to equation (1), the best query in case of multi class uncertainty sampling is given by

$$x^*_M = \text{argmin}\ P_\theta(y_1 \mid x) - P_\theta(y_2 \mid x)$$

where y_1 and y_2 are the first and second most probable class labels. Accordingly, the querying function for the entity and relation detection stage of information extraction can be given as

$$q_{ERD} = \text{argmin}\ p(y \mid x_i) - p(y' \mid x_i)$$

or

$$q_{ERD} = \text{argmin}\ [f(x_i, y) - f(x_i, y')]$$

i = 1 to n and y and y′ are the first and second most probable class labels.

For all the stages, the performance would be calculated using three metrics i.e. precision, recall and F-measure. For POS tagging, precision would be calculated as number of correctly retrieved tags divided by the total number of retrieved tags. Recall would be calculated as number of correctly retrieved tags divided by the actual number of tags. For entity detection, precision would be calculated as the number of correctly extracted entities divided by the total number of extracted entities and recall would be calculated as number of correctly extracted entities divided by the actual number of entities. For relation extraction, precision would be calculated as the number of correctly extracted relations divided by the total number of extracted relations and recall would be calculated as the number of the correctly extracted relations divided by the actual number of relations. F- Measure for all these stages is equal to 2*precision*recall / precision + recall.

6. Conclusion and Future Work

. In this paper we discussed an active learning process for the pipelining of information extraction with focus on including part-of-speech tagging stage into the pipeline. In Section 5.1 we theoretically showed how active learning can be applied to part-of-speech tagging and included into the pipeline. In future we intend to show its empirical implementation and performance evaluation using the above mentioned metrics.

7. Acknowledgement

The authors are thankful to the faculty, Department of Computer Science, University of Kashmir for their constant support.

8. References

1. Sunita, S., and Anuradha, B. 2002. "Interactive Deduplication using Active Learning". *In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD-2002)*, Edmonton, Canada.

2. Roth, D. And Small, K. 2008. "Active learning for Pipeline Models". *AAAI 2008*, pp. 683-688.

3. Bunescu, R. C., and Mooney, R. J. 2005. "A Shortest Path Dependency Kernel for Relation Extraction". *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, ACL, 724-731.

4. Razvan, B. 2008. "Learning with Probabilistic Features for Improved Pipeline Models". *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, 670–679.

5. Sunita, S. 2007. "Information Extraction". *Foundations and Trends in Databases* 1(3): 261–377.

6. Kevin, M., Michael, B., Jules, B., Wendy, C., John, G., Dilip, G., James, H., and Elizabeth, L. 2004. "Implementation and Evaluation of a Negation Tagger in a Pipeline-based System for Information Extraction from Pathology Reports". MEDINFO, 663-667.

7. Steven, B., Ewan, K., and Edward, L. 2006. "Natural Language Processing/ Computational Linguistics with Python".

8. Finkel, J. R.; Manning, C. D.; and Ng, A. Y. 2006. "Solving the problem of cascading errors: Approximate Bayesian inference for linguistic annotation pipelines". *In Proc. Of the Conference on Empirical Methods in Natural Language Processing* (EMNLP).

9. Manish, A., Ajay, G., Rahul, G., Prasan, R., Mukesh, M., and Zenita, I. 2007. "Liptus: Associating structured and unstructured information in a banking environment". *Proceedings of the 2007 ACM SIGMOD*, 915-924.

10. Xiaofeng, Y., and Wai, L. 2010. "Bidirectional Integration of Pipeline Models". *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, 1045-1050.

11. Chang, M.-W.; Do, Q.; and Roth, D. 2006. "Multilingual dependency parsing: A pipeline approach". *In Recent Advances in Natural Language Processing*, 195–204.

12. Jordi, T., Alicia, A., and Neus, C. 2006. Adaptive Information Extraction, ACM Computing Surveys, 38(2).

13. Matthew, M., and Craig, K. 2005. "Semantic annotation of unstructured and ungrammatical text". *In Proceedings of the 19th International Joint Conference on Artificial Intelligence* (IJCAI), 1091–1098.

14. Shubin, Z., and Ralph, G. 2005. "Extracting relations with integrated information using kernel methods". *Proceedings of the 43rd Annual Meeting On Association for Computational Linguistics*, 419-426.

15. Henning, W., Benno, S., and Gregor, E. 2011. "Constructing Efficient Information Extraction Pipelines". *CIKM'11 ACM*, Scotland, UK.

16. Nguyen, B., and Sameer, B. "A Review of Relation Extraction". Language Technologies Institute, School of Computer Science Canergie Mellon University, Pittsburgh.

17. Nilsson, N.J. "Introduction to Machine Learning". Department of Computer Science, Stanford University.

18. Keigo, W., Danushka, B., Yutaka, M., and Mitsuru, I. 2009. "A Two-Step Approach to Extracting Attributes for People on the Web". *ACM*, Madrid, Spain.

19. Huma, L., Craig, S., John, S-T., Nello, C., and Chris, W. 2002. "Text Classification Using String Kernels". *Journal of Machine Learning Research*, 419-444.

20. Burr, S. 2010. "Active Learning Literature Survey", Computer Sciences Technical Report 1648, University of Wisconsin–Madison.

21. Thompson, C.A., Califf, M.E., and Mooney, R.J. "Active Learning for Natural Language Parsing and Information Extraction". *In Proceedings of the Sixteenth International Machine Learning* Conference,406-414.

22. T. Scheffer, C. Decomain, and S.Wrobel. 2001. "Active hidden Markov models for information extraction". *In Proceedings of the International Conference on Advances in Intelligent Data Analysis*, Springer-Verlag, 309-318.

23. D. Lewis and W. Gale. 1994. "A sequential algorithm for training text classifiers". *In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval. ACM/Springer*, 3-12.

24. A. Culotta and A. McCallum. 2005. "Reducing labeling effort for stuctured prediction tasks". *In Proceedings of the National Conference on Artificial Intelligence* 746–751.

25. Burr, S. 2009. "Active Learning. Advanced Statistical Language Processing". Machine Learning Department, Carnegie Mellon University.

26. H.S. Seung, M. Opper, and H. Sompolinsky. "Query by committee". *In Proceedings of the ACM Workshop on Computational Learning Theory*, 287–294.

27. Nanda, K. 2004. "Combining Lexical, Syntactic, and Semantic Features with Maximum Entropy Models for Extracting Relations". *Proceedings of the ACL* 2004.