Department of Mathematics and Computer Sciences
University of Science, Vietnam

Punctuation Prediction for Vietnamese Texts Using Conditional Random Fields

by

Pham Hong Quang

Supervisors:

Nguyen Thanh Binh
Department of Computer Science
University of Science, Vietnam

Nguyen Viet Cuong
Department of Computer Science
National University of Singapore

07–2015

Abstract

We present a novel approach for the punctuation prediction problem in Vietnamese language. It is an important task since it can be used to add appropriate punctuations to machine-transcribed speeches which usually lack such information. Similar to previous works for English language and Chinese language, we solve the problem by using the conditional random field model and propose a set of useful features for punctuation prediction. We illustrate the approach by experimental results along with a corpus collected from movies subtitles.

Keywords: Punctuation prediction, Vietnamese, conditional random field

Acknowledgement

I would like to express my deepest and sincere gratitude to my thesis advisors, Nguyen Thanh Binh and Nguyen Viet Cuong, for their guidance and priceless advices during my thesis.

I would love to give my thankfulness to Professor Pham The Bao for his fruitful support. I also want to appreciate all my friends at Hochiminh University of Science for their help and discussions during my 4 years of study.

Finally and most importantly, I would like to extend my deepest appreciation to my family, my parents and my older brother, for their love, encouragement and being my sheet-anchor.

Pham Hong Quang

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we discuss the punctuation prediction problem for Vietnamese language. We begin with our motivation and then give a brief overview for the problem. The outline of the thesis is described in the last section.

1.1 Motivation

Punctuation prediction is an important problem in language and speech processing, especially with the rapid growth of multimedia technology [Beath et al., 2012]. Models from punctuation prediction systems can be used to annotate machine-transcribed speech texts which usually do not come with punctuations, e.g. automatic captions from speeches. [1]

Although this task can be done by human, it is not efficient considering the colossal amount of data we can get from videos uploaded to the internet everyday. Captions from those videos provide a rich source of data for language processing. Therefore, there is a need to build a system that can assign punctuations to machine-transcribed speech from videos and provides better quality data for later applications.

In this work, we model this problem as a sequence labeling task: each word in a sentence is labeled by a punctuation. To do this, we add an empty punctuation into

[1] An example is available at `https://www.google.com/intl/en/chrome/demos/speech.html`

the set of labels to indicate that the word corresponding to this label is not followed by any conventional punctuations. Among many sequence labeling models such as *Hidden Markov Models* (HMMs) [Rabiner, 1989a], *Maximum Entropy Markov Models* (MEMMs) [McCallum et al., 2000], etc., each suffers from certain weaknesses that make them not applicable for this particular task. For example, HMMs are generative models, so they model the joint distribution over both input and output sequences. However, in our work, we only concern about the conditional probability over the output sequence.

In contrast to HMMs and MEMMs, Conditional Random Fields (CRFs), especially linear-chain CRFs [Lafferty et al., 2001] are especially designed for the sequence labeling problem and then applied to other fields with promising results. CRFs are discriminative models, which assume the data are given; therefore, they allow us to define a rich set of features to capture complex dependencies between the input and output sequences. One can see more details of this model in Chapter 2.

This problem has been extensively researched for major languages such as English language [Beeferman et al., 1998, Huang and Zweig, 2002, Lu and Ng, 2010, Cuong et al., 2014] and Chinese language [Lu and Ng, 2010, Zhao et al., 2012]. However, to the best of our knowledge, punctuation prediction has not been investigated for the Vietnamese language.

1.2 Related Works

Punctuation prediction has been broadly investigated for many years. One of the first punctuation prediction systems was created by Beeferman et al. [1998] to automatically insert commas into texts. The system uses a finite state transition model and Viterbi decoder to predict the positions of commas in a sentence. Huang and Zweig [2002] proposed a maximum entropy model for the task with 3 punctuations: period, comma, and question mark.

Using CRF models, Lu and Ng [2010] can achieve better performances for punctuation prediction task on both the English and Chinese data sets of the IWSLT corpus [Paul, 2009]. Notably, they showed that using a dynamic CRF to jointly model

word-level and sentence-level labeling tasks and thus capture some long-range dependencies is useful for punctuation prediction. Similarly, Cuong et al. [2014] used high-order semi-Markov CRFs to capture long-range dependencies between punctuations and achieved better prediction performance than linear-chain CRFs. Zhao et al. [2012] investigated Chinese punctuation prediction by formulating the problem as a multiple-pass labeling task and applying the CRF model. Cho et al. [2012] studied a segmentation and punctuation prediction problem for German-English with a monolingual translation system and demonstrated their results in the oracle experiments.

Our work is one of the Vietnamese language processing tasks. Related to Vietnamese language processing, there have been various works in different directions such as word segmentation [Dien et al., 2001, Nguyen et al., 2006] and part-of-speech (POS) tagging [Tran et al., 2009]. Using a weighted finite state transducer and neural network, Dien et al. [2001] built a Vietnamese word segmentation system with high precision. Nguyen et al. [2006] also investigated the Vietnamese word segmentation problem using CRF and SVM models. The Vietnamese POS tagging task was studied by Tran et al. [2009] with three different techniques: maximum entropy model, CRF, and SVM.

1.3 Outline of The Thesis

In this thesis, we build a system that can automatically assign punctuations to sentences from machine-transcribed documents to provide a better text quality for further applications.

The thesis is organized as follows. In Chapter 1, we introduce the problem and related works. The main method used in our approach is given in Chapter 2. In Chapter 3, we describe our proposed approach for the Vietnamese punctuation prediction task. We illustrate our techniques by experimental results in Chapter 4. The thesis ends with conclusions and future improvements.

Chapter 2

Preliminaries

In this chapter, we introduce *graphical models* and *linear-chain conditional random field*, a typical type of undirected graphical models that is widely used in practice for predicting a sequence of labels given a sequence of observations. We also give an introduction to the task of *part-of-speech* tagging, a concrete example where CRF models are applied.

2.1 Graphical Representation

We consider probability distributions over a set of random variables $V = X \cup Y$ where X is the set of input variables that we assume to be observed. Similarly, Y is the set of output variables that we wish to predict. Each random variable V_i is in general a vector in a vector space. In practice, there are several inquiries we want to make regarding such variables. For example, we might be interested in knowing whether two subsets of variables are independent from each other or whether they are conditionally independent given a third set. Another kind of inquiry that leads to the invention of CRF is computing the conditional probabilities: the probability of one subset of variables given the values of another subset. These inquiries can be answered if we have the joint distribution of the variables $p(X \cup Y)$. Thus, a graphical model can be interpreted as a model that manipulates the probabilities between variables and can answer inquiries about values of sets of random variables. In the scope of this thesis, we only consider how to compute the conditional probability

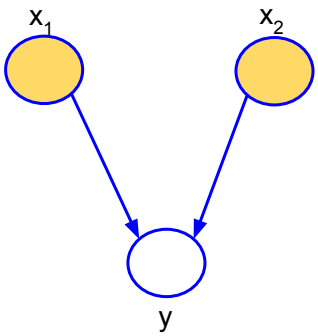


Figure 2.1. Independency graph

using graphical models.

A probabilistic graphical model is represented as a graph [Koller and Friedman, 2009]. In this graph, there is a node for each random variable and the absence of an edge between any two nodes represents the conditional independence between the two corresponding variables. Conditional independence means that two random variables a and b are independent given a third random variable c if given knowledge of c occurring, the knowledge of a occurring provides no information on the likelihood of b occurring. Formally, a and b are conditional independence given c if $p(a, b|c) = p(a|c)p(b|c)$, or equivalently $p(b|a, c) = p(b|c)$ where $p(a|c)$ denotes the conditional probability of a given c, and $p(a, c)$ denotes the joint probability of a and c.

Conditional independence is an important concept because it can be used to decompose a complex probability distribution into a product of factors, each of which consists a subset of corresponding variables. One can find an example in Section 2.1.1. Note that a graph with fully connected nodes provides no information about the underlying probability distribution.

Figure 2.1 shows an example of an independency graph. In this graph, the vertices x_1 and x_2 are the input or observation variables (shaded circles), and y is the output variable (empty circle).

As an example, Figure 2.1 shows an independency graph that has the underlying probability distribution $p(x_1, x_2, y)$ which can be factorized as:

$$p(x_1, x_2, y) = p(x_1)p(x_2)p(y|x_1, x_2). \tag{2.1.1}$$

Here, the corresponding factors are $\Psi_1(x_1) = p(x_1),\ \Psi_2(x_2) = p(x_2),$

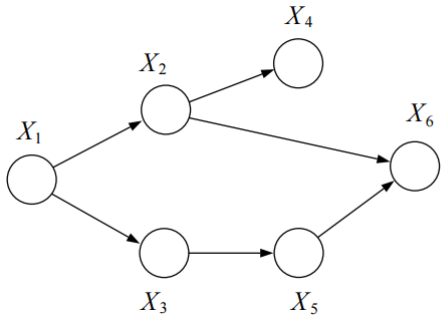


Figure 2.2. An example of a directed graphical model.

$\Psi_3(y) = p(y|x_1, x_2)$

There are two main types of probabilistic graphical models: directed and undirected models corresponding to the directed and undirected graphs in their representation. In the following sections, we give a brief introduction about these models.

2.1.1 Directed Models

Directed graphical models (also called Bayesian Networks) are represented by directed acylic graphs, i.e. all edges in the graph are directed and no cycles are allowed. In directed graphical models, if there is a directed edge from node v_i to node v_j, then v_i is a parent of v_j. A probability distribution $p(v)$, $v \in V$ of a directed graphical models can be written as a product of conditional distribution of each node v_k that is only conditioned on the parent nodes $\pi(v_k)$:

$$p(v) = \prod_k p(v_k|\pi(v_k)), \tag{2.1.2}$$

which means the node v_k is independent to all other nodes given its parents.

In Figure 2.2, we show an example of a graphical model over 6 nodes. One can compute the joint probability over these nodes using the *chain rule* as:

$$p(x_1, \ldots, x_6) = p(x_1)p(x_2|x_1)p(x_3|x_2, x_1) \ldots p(x_6|x_5, \ldots, x_1). \tag{2.1.3}$$

This factorization is always correct regardless of the conditional independence between the nodes. By applying the Formula 2.1.2, we obtain the joint probability for Figure 2.2 as follows:

$$p(x_1, \ldots, x_6) = p(x_1)p(x_2|x_1)p(x_3|x_1)p(x_4|x_2)p(x_5|x_3)p(x_6|x_2, x_5). \tag{2.1.4}$$

Using this property of directed models allows us to simplify several terms in the chain rule and represent the joint probability in a compact form. This leads to a saving in both time to calculate the probability and the space to represent the nodes. The savings in this example are minimal but in general when the number of nodes is large, the saving in time and space will become impactful.

2.1.2 Undirected Models

In contrast to directed graphical models, undirected graphical models (also called Markov Random Fields) are represented by undirected graphs and can contain arbitrary cycles. In this model, the probability distribution $p(v)$ is factorized over the maximal cliques $\mathcal{C}$ of the graph. A clique is a subset of fully connected nodes and a maximal clique is a clique that no node can be added to create a new clique. Then, we can factorize the graph into cliques: conditional independent nodes do not appear within the same factor, that means they belong to different cliques:

$$p(v) = \frac{1}{Z} \prod_{c \in C} \Psi_c(v_c). \tag{2.1.5}$$

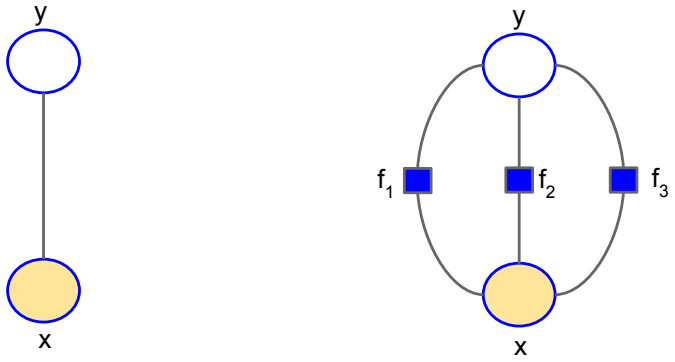
The factors $\Psi_C \geq 0$ are also called potential functions of the random variables v_c within a clique $c \in \mathcal{C}$.

In undirected graphical models, a potential function will assign a real value to a subset of the random variables v in a clique. Note that the potential functions do not necessarily have a probabilistic interpretation. An example of the potential functions will be given in Section 2.2.2. Figure 2.3 shows an undirected graphical models of two nodes. The edge between these nodes is represented by using three potential functions as shown in Figure 2.3b.

Since the potential functions do not necessarily have a probabilistic interpretation, normalization of the product of potential functions is necessary to achieve a proper probability measure. This is yielded by a normalization factor

$$Z = \sum_{v} \prod_{c \in \mathcal{C}} \Psi_c(v_c) \tag{2.1.6}$$

in Equation 2.1.5. The main difference between directed and undirected graphical models is the way the original distribution is factorized. In directed graphical models,



(a) Independency graph

(b) A graph represents an edge using three potential functions

Figure 2.3. An example of an undirected graphical model

the distribution is factorized into a product of conditional probability distributions. On the other hand, in undirected graphical models a factorization into arbitrary functions is achieved. This does not require an explicit specification on how the variables are related.

2.2 Linear-chain Conditional Random Fields

2.2.1 Introduction

Conditional random field, which is a type of undirected models, was introduced by Lafferty et al. [2001]. It was especially designed to overcome several weaknesses of previous models such as HMM and MEMM. For example, CRF is a discriminative model, which learns a conditional distribution, so it does not have to model the dependencies between the observations. This allows us to incorporate more complex and useful features over the observations and its label sequence, which is hard to accomplish with HMM. CRF also overcomes the label bias problem encountered when using MEMM [Lafferty et al., 2001]. In this section, we introduce CRF in general and linear-chain CRF, a special type of CRFs that is widely used in practice.

To formally define a CRF, let $\mathbf{X}$ and $\mathbf{Y}$ be random vectors, $\mathbf{x}$ be a value of $\mathbf{X}$, and $\mathbf{y}$ be a value of $\mathbf{Y}$ such that $\mathbf{x} = (x_1, x_2, \ldots, x_T)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_T)$. A

CRF in general can be derived from Formula 2.1.5:

$$p(\mathbf{v}) = \frac{1}{Z} \prod_{c \in C} \Psi_c(\mathbf{v_c}) \tag{2.2.7}$$

where $\mathbf{v}$ is an arbitrary random vector. Then, the conditional probability $p(\mathbf{y}|\mathbf{x})$ can be written as

$$p(\mathbf{y}|\mathbf{x}) = \frac{p(\mathbf{x}, \mathbf{y})}{p(\mathbf{x})} \tag{2.2.8}$$

$$= \frac{p(\mathbf{x}, \mathbf{y})}{\sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z})} \tag{2.2.9}$$

$$= \frac{\frac{1}{Z} \prod_{c \in C} \Psi_c(\mathbf{x}_c, \mathbf{y}_c)}{\frac{1}{Z} \sum_z \prod_{c \in C} \Psi_c(\mathbf{x}_c, \mathbf{z}_c)}. \tag{2.2.10}$$

From this, the general formulation of CRF is derived:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{c \in C} \Psi_c(\mathbf{x}_c, \mathbf{y}_c), \tag{2.2.11}$$

where Ψ_c are the different factors in the decomposition as mentioned in Section 2.1.2 and $Z(\mathbf{x}) = \sum_z \prod_{c \in C} \Psi_c(\mathbf{x}_c, \mathbf{z}_c)$ is the normalization function, also called the partition function.

Note that a CRF models the conditional distribution not the joint distribution, so it can answer inquiries about calculating the conditional probabilities efficiently. Other queries such as independence between subsets of variables cannot be answered easily with CRF since we have to properly marginalize over subsets of variables.

2.2.2 Linear-chain CRF

A CRF in general can take an arbitrary graph structure. For the problem of punctuation prediction, we only consider linear-chain CRF, a particular class of CRFs. Linear-chain CRF is restricted to a special type of graphs by making an assumption that the data can be modelled as a sequence of labels with the first order Markov assumption and the observations (also in form of a sequence) given to the model.

Figure 2.4 shows a graphical representation of a linear-chain CRF where the shaded node depicts a sequence of observations and the white nodes are the labels. Edges between white nodes describe the first order Markov assumption over the label.

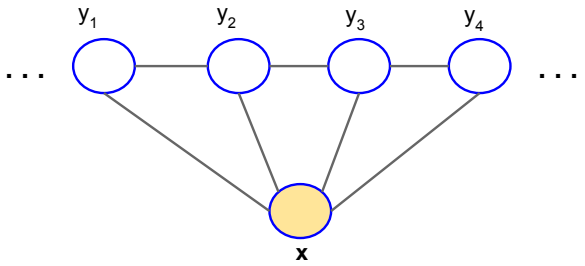


Figure 2.4. Graphical representation of a linear-chain CRF

Linear-chain CRF defines the conditional distribution $p(\mathbf{Y}|\mathbf{X})$ that has the following form

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left(\sum_{k=1}^{K}\sum_{t=1}^{T} \lambda_k f_k(y_t, y_{t-1}, \mathbf{x}_t)\right), \tag{2.2.12}$$

again, $Z(\mathbf{x}) = \sum_{\mathbf{y}} \exp\left(\sum_{k=1}^{K}\sum_{t=1}^{T} \lambda_k f_k(y_t, y_{t-1}, \mathbf{x}_t)\right)$ is the normalization function. In linear-chain CRF, the factor $\Psi_c(\mathbf{x}_c, \mathbf{y}_c)$ is defined as

$$\Psi_c(\mathbf{x}_c, \mathbf{y}_c) = \sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, \mathbf{x}) \tag{2.2.13}$$

where $f_k(y_t, y_{t-1}, \mathbf{x})$ is a potential function, usually called features, and $\{\lambda_k\}_1^K$ are the weights that we wish to learn. A feature of CRF, in general, is a real-valued function over the observation $\mathbf{x}$, the position t and two labels y_t, y_{t-1}. For example, one can consider the following observation and its label:

Sentence: Tôi đi học

Label: O O PERIOD

(I go to school.)

and a feature template:

$$f_{\text{word,tag}}(y_t, y_{t-1}, \mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x}_t = \text{word and } y_t = \text{tag} \\ 0 & \text{otherwise} \end{cases}. \tag{2.2.14}$$

Applying this template to the last position of the above observation, i.e. $t = 3$, gives the following feature:

$$f_1 = f_{\text{học,PERIOD}}(y_3, y_2, \mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x}_3 = \text{"học" and } y_3 = \text{"PERIOD"} \\ 0 & \text{otherwise} \end{cases}.$$

In this case, the word "học" and the label "PERIOD" are the feature template's parameters and their values vary as the position t moves along the observations. In this example, the above feature f_1 is only equal to one if it is applied at the last word of the sentence, otherwise, its value is zero.

To create all the features given a data set, first, we define the set of feature templates as (2.2.14). For each template, by moving the position t all over the observation $\mathbf{x}$, we generate all the features according to that template. Then, we can repeat the process with another template until all features are generated. By changing the set of feature templates, we can adjust the features we want to express from the data.

For simplicity, we drop the parameters and assume that there are K features $f_1, f_2, \ldots, f_K$. Since the size of the training data is usually large and we also might want to use multiple features templates, this could lead to a very large number of features generated.

The use of features in CRF provides a way to exploit different aspects of the observation. For example, in *named entity recognition* [Grishman and Sundheim, 1996], an HMM relies only on the word's identity. However, it may encounter many unseen words during testing such that relying only on the word's identity is not sufficient. Instead, we may want to exploit other features of a word such as its word form and its position in a sentence, or we may even build several dictionaries – one for person names, one for location names, etc. for a specific task. CRF allows us to incorporate such characteristics we want to express easily in terms of features because it assumes that the data are given. In contrast, doing the same thing with models like HMMs may lead to intractability [Sutton and McCallum, 2006].

2.2.3 Parameters Estimation

Given the training data $\mathcal{D} = \{(\mathbf{x}^i, \mathbf{y}^i)\}_i^N$, the CRF model is trained by choosing the parameters $\vec{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_K)$ that maximize the following conditional log-likelihood of the data:

$$\begin{aligned}\mathcal{L}(\mathcal{D}) &= \sum_{(\mathbf{x},\mathbf{y}\in\mathcal{D})} log\, p(\mathbf{y}|\mathbf{x}) \\ &= \sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{t=1}^{T}\lambda_k f_k(y_t^i, y_{t-1}^i, \mathbf{x}_t^i) - \sum_{i=1}^{N} logZ(\mathbf{x}^i),\end{aligned} \tag{2.2.15}$$

where λ_i is the parameters we need to learn.

Since we are dealing with a large number of parameters, there would be some vectors whose norm is too large and dominant others, which is called overfitting. To avoid this, the likelihood is penalized by the term $\sum_{i=1}^{n}\frac{\lambda_1^2}{2\sigma^2}$ where σ is a parameter that controls the degree of regularization. By adding the regularization terms, the large norm vectors are penalized and the chance that there are some dominant weights occurring will be reduced. Finally, we want to maximize the regularized log likelihood:

$$\mathcal{L}(\mathcal{D}) = \sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{t=1}^{T}\lambda_k f_k(y_t^i, y_{t-1}^i, \mathbf{x}_t^i) - \sum_{i=1}^{N} logZ(\mathbf{x}^i) - \sum_{k=1}^{K}\frac{\lambda_k^2}{2\sigma^2}. \tag{2.2.16}$$

In general, the maximum of $\mathcal{L}(\mathcal{D})$ cannot be computed in closed form, therefore, numerical optimization is used. Quasi-Newton methods BFGS [Bertsekas, 1999] or LBFGS [Liu et al., 1989] are applied to find the optimum of $\mathcal{L}(\mathcal{D})$. The partial derivatives of 2.2.16 are:

$$\frac{\partial\mathcal{L}}{\partial\lambda_k} = \sum_{1=1}^{n}\sum_{t=1}^{T}\sum_{k=1}^{K}\lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, x_t^{(i)}) - \sum_{i=1}^{N} logZ(\mathbf{x}^i) - \sum_{k=1}^{K}\frac{\lambda_k}{2\sigma^2}. \tag{2.2.17}$$

The first term of the derivation can be calculated by counting how often each feature occurs in the training data [McDonald and Pereira, 2005]. Calculating the second term requires a dynamic programming approach. This can be done by applying the variant of the forward-backward algorithm [McDonald and Pereira, 2005]. Finally, note that the function $\mathcal{L}(\lambda)$ is concave because of the convexity of functions of the form $g(x) = log\sum_i \exp x_i$. Moreover, adding the regularization term makes the function strictly concave, which ensures that there is only one local optimum and it is also the global optimum.

2.2.4 Inference

After training a CRF model, we want to use it to label a new observation $\mathbf{x}$. One problem during inference is how we can find the label sequence $\mathbf{y}^*$ that satisfies $y^* = \text{argmax}_y \ p(\mathbf{y}|\mathbf{x})$. Trying every possible value of $\mathbf{y}$ is practically impossible because of the large number of possible tag sequences. In linear-chain CRF, the Viterbi Algorithm [Rabiner, 1989b] is applied to find the most likely sequence label given an observation. The Viterbi Algorithm is similar to the Forward-Backward Algorithm. The main difference is that instead of summing, a maximization is applied.

Let S be the label space and $s \in S$ be a state. We define the quantity $\delta_j(s|\mathbf{x})$ as the probability of the most probable path ending in position j at state s, i.e.,

$$\delta_j(s|\mathbf{x}) = \max_{y_1,y_2,\ldots,y_{j-1}} p(y_1, y_2, \ldots, y_j = s|\mathbf{x}). \tag{2.2.18}$$

The induction step is:

$$\delta_{j+1} = \max_{s' \in S} \delta_j(s').\Psi_{j+1}(\mathbf{x}, s, s'). \tag{2.2.19}$$

Let $\psi_j(s)$ be the array that keeps track of the j and s values. Then the algorithm works as follows:

1. Initialization:

   We initialize the values for all steps from the starting state s_0 to all possible first states s by setting the corresponding factor values:

   $$\forall s \in S: \quad \delta_1(s) = \Psi_1(\mathbf{x}, s_0, s)$$
   $$\psi(s) = s_0.$$

   The starting state s_0 indicates the start of a sequence.

2. Recursion:

   The values for the next steps are computed from the current value and the maximum values regarding all possible succeeding states s':

   $$\forall s \in S: \ 1 \leq j \leq n : \delta_j(s) = \max_{s' \in S} \delta_{j-1}(s')\Psi(\mathbf{x}, s', s)$$
   $$\psi(s) = \text{argmax}_{s' \in S} \ \delta_{j-1}(s')\Psi(\mathbf{x}, s', s).$$

3. Termination:

$$p^* = \max_{s' \in S} \delta_n(s')$$
$$y_n^* = \operatorname*{argmax}_s \ \delta_n(s').$$

4. Backtracking:

$$y_t^* = \psi_{t+1}(y_{t+1}^*) \quad t = n-1, n-2, \ldots, 1.$$

After step 3, the array $\psi_i(s)$ is filled with the optimal path to reach the final state p^*, and step 4 reads the best path to that state which results in the most likely sequence that the model predicts.

2.3 Part of Speech Tagging

A part of speech (POS) is a type of a words which have similar grammatical properties. Words that have the same POS generally share similar behaviors in terms of syntax, which means they play similar roles within the grammatical structure of sentences. Several examples on POS in English are: [1] noun, pronoun, verb, etc. Part of speech tagging (also called POST) is a task of assigning a POS to every word in a sentence. For example, the following sentence will be assigned POS as:

> Nhiều chuyên gia muốn Apple thu hồi iPhone 4
> Nhiều/A chuyên/A gia/V muốn/V Apple/Np thu/Np hồi/N iPhone/A 4/M
> (Many experts suggest Apple should recall iPhone 4),

where the tag A means "Adjective", N means "Noun", V means "Verb", Np means "Personal noun" and M means "Numeral".

Table 2.1 shows the list of POS as well as their explanation in both English and Vietnamese, according to [Nguyen and Phan, 2007]. POST is a typical problem

[1] List of full POS for English can be found at `https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html`

	POS	Explanation in English	Explanation in Vietnamese
1	N	Noun	danh từ
2	Np	Personal Noun	danh từ riêng
3	Nc	Classification Noun	danh từ chỉ loại
4	Nu	Unit Noun	danh từ đơn vị
5	V	Verb	động từ
6	A	Adjective	tính từ
7	P	Pronoun	đại từ
8	L	Attribute	định từ
9	M	Numeral	số từ
10	R	Adjunct	phụ từ
11	E	Preposition	giới từ
12	C	Conjunction	liên từ
13	I	Interjection	thán từ
14	T	Particle, Modal Particle	trợ từ, tiểu từ
15	B	Words from foreign countries	từ mượn nước ngoài
16	Y	Abbreviation	từ viết tắt
17	X	Unknown Words	các từ không phân loại được
19	Mrk	Punctuations	dấu câu

Table 2.1. List of POS for Vietnamese

which can be solved using linear-chain CRF [Tran et al., 2009]. In this work, we consider POS of a word as extra information for the task of punctuation prediction. However, since existing models for this task are built using information that is not available during punctuation prediction such as capitalisation or the punctuation itself, we demonstrate that applying those models directly results in poor performance.

Chapter 3

Punctuation Prediction for Vietnamese Texts using CRFs

In this chapter, we introduce our approach to apply CRF to the task of punctuation prediction for the Vietnamese language. We first describe our corpus in Section 3.1, this will be followed by our approach in Section 3.2 and the set of features for CRF in Section 3.3.

3.1 Corpus

To create a suitable corpus for the punctuation prediction task, we build a corpus from 60 Vietnamese subtitles. These subtitles consist of conversational speeches from online movies and they are manually annotated by human. As a pre-processing step, we clean the data by fixing common writing errors or non-standard uses of punctuations like two or more punctuations at the end of a sentence. After the pre-processing step, we end up with a corpus of about 7200 sentences with around 580,000 words. Because the corpus is built from speeches, we only consider 4 types of punctuations to predict for this corpus: comma (,), period (.), question mark (?) and exclamation mark (!).

3.2 Punctuation Prediction as Sequence Labelling

Like previous works for English and Chinese [Lu and Ng, 2010, Zhao et al., 2012], we model the punctuation prediction task as a sequence labelling problem. In particular, we treat each sentence in our corpus as a sequence and aim to label each word in the sequence by a punctuation that immediately follows the word. In the simple case, we use the label "O" to indicate that a word is not followed by any punctuation. For example, considering the following sentence in Vietnamese: [1]

> Tôi nghĩ ông làm được mà, ông Morrison.
>
> (I think you can do it, Mr. Morrison.),

this sequence can be labelled as follows. We note that all the words are in lower case because the word case information is usually not available for the punctuation prediction task. For instance, when the texts are transcribed from speeches, we do not have the case information for the words.

> tôi/O nghĩ/O ông/O làm/O được/O mà/COMMA ông/O morrison/PERIOD

3.3 Features for CRFs

We construct a set of features that are useful for the punctuation prediction task. First we start with the orthographic features: the current word and the current label. After that, we add more word features to the set, this time, we use words in a window of size 5 relative to the current position. We also add the transition label feature to better describe the context of the sentence. Each time we add new features to the set, we retrain and test the model, if the performance of the new model does not increase, it means the recently added features are not helpful and they are removed from the set. We also add two position features: distance of the current word from the beginning of the sequence and to the end of the sequence.

[1] The sentence is extracted from the movie Cat's Eye (1985) `http://www.imdb.com/title/tt0088889/`

These features help to capture dependencies between length of a sentence and its ending punctuation as well the position of commas inside that sentence.

The last type of features we use is the part-of-speech (POS) features. To get the POS of each word (in our experiment, we use the term "word" to refer to a syllable since we do not segment the sentences into words), we use the CRF model from JVnTextPro [Nguyen et al., 2005]. Although the POS tagger was built using information about punctuation and capitalisation to correctly predict the POS, in the task of punctuation prediction, such information is not available. In our experiment, we use the tagger directly without any modification.

Table 3.1 details our set of features. These features are categorized into three groups: position features, orthographic features, and context/POS features as we described above.

Position Features:
- Position from the beginning of the sequence.
- Position to the end of the sequence.
Orthographic Features:
- Identity of the current word.
- The current label.
Context Features:
- Identity of the previous word relative to the current word.
- Identity of the word at -2^{th} position relative to the current position
- Identity of the word at $+2^{th}$ position relative to the current position
- Identity of the word at -3^{th} position relative to the current position
- Identity of the word at -4^{th} position relative to the current position
- Identity of the next word relative to the current word.
- Identity of the current word and the previous word.
- Identity of the current word and the preceding word.
- Identity of two previous words.
- Identity of two preceding words.
- Transition between labels.
- POS of the current word.
- Identity and POS of the current word.

Table 3.1. Set of features for the CRF model.

Chapter 4

Experiments

The previous chapters have outlined our approach to the problem of punctuation prediction in the Vietnamese language. This chapter presents our results of the CRF models. We also discuss a few cases where the models failed to get the right punctuations. Finally, we give our discussion about the performance of the model.

4.1 Evaluation Metrics

We use three metrics to measure the performance of our system: precision (denoted by P), recall (denoted by R), and F_1 score (denoted by F). From Table 4.1, the punctuations are not equally distributed, hence we use micro-averaged scores [D. Christopher et al., 2008] instead of macro-averaged scores for the overall performance of the system. The micro-averaged precision and recall formulas are given as follows:

$$P = \frac{\sum_j tp_j}{\sum_j (tp_j + fp_j)} \qquad\qquad R = \frac{\sum_j tp_j}{\sum_j (tp_j + fn_j)}, \tag{4.1.1}$$

where tp_j is the number of documents correctly classified as class j (true positive), fp_j is the number of documents incorrectly classified as class j (false positive), and fn_j is the number of documents in class j that are misclassified (false negative). The micro-averaged F_1 score is computed as

$$F = \frac{2PR}{P + R}. \tag{4.1.2}$$

Punctuation	Training set		Testing set	
	Number	Percentage (%)	Number	Percentage (%)
Period	3656	13.67	1818	13.61
Comma	1238	4.63	590	4.41
Question mark	1006	3.75	511	3.82
Exclamation mark	299	1.11	164	1.22

Table 4.1. Distribution of punctuations in training and testing data sets. Note that the rest of the data sets contain no punctuation (label O).

4.2 Experiments Setup

For the experiments, we split our corpus into two parts: 70% for training and the rest for testing. Table 4.1 summarizes the punctuations distribution over the training and testing sets. In our data sets, comma and period are the most common punctuations, while question marks and exclamation marks are very rare.

In our experiments, we illustrate the effects of different combinations of features to the performance of our Vietnamese punctuation prediction system. We begin with the orthographic features and subsequently add the position and context features. Finally, we add the POS features to build a model that uses all features proposed. We train CRF models on the training set with each token being a word and the regularizer $\sigma = 1$, and then test our models by the testing set. Our scores are computed on the token level.

4.3 Results

In Table 4.2, we show the results of different combinations of features for our Vietnamese punctuation prediction system where the highest precision, recall and F_1 scores are shown in bold. Using orthographic features alone achieves 26.74% F_1 score while adding position features increases F_1 score to 75.96%, an increase of 49.22%. Adding context features increases the F_1 score to 78.86%, which is also

Features	P (%)	R (%)	F (%)
Orthographic	56.54	17.51	26.74
Positoin + Orthographic	84.12	69.25	75.96
Position + Orthographic + Context	**85.86**	**72.91**	**78.86**
Position + Orthographic + Context + POS	73.76	67.59	70.54

Table 4.2. Results for different features combinations.

the highest performance with linear-chain CRF models we achieved so far. Notably, adding POS features does not increase F_1 score of the models. This is mainly because punctuations and capitalisation, each of which is the input to the POS model, are missing in this task and it makes the POS tagger become less accurate. This case shows one of the main challenges in the punctuation prediction problem: how to incorporate information such as POS to the punctuation prediction system without hurting the performance.

In Table 4.3, we show a comparison between a model trained with three types of features: position, orthographic and context features (denoted by CRF_3) and a model trained with all types of features, including the part of speech (denoted by CRF_{pos}). Adding POS features to CRF_3 worsens the scores in all of the punctuations except the exclamation marks. Our best model, CRF_3, achieves promising results on three punctuations: comma, period and question mark. Notably, our models still perform poorly in predicting the exclamation marks; however, it does not affect the overall score significantly since this label's frequency in the dataset is quite small.

4.4 Error Analysis

In this section, we give an investigation on the output of CRF model and show several cases where they fails to predict the correct output. In this task, we choose the model CRF_3 in Table 4.3. From our analysis, there are two main cases that the model predicts the punctuations incorrectly.

Punctuation	CRF_{pos}			CRF_3		
	P (%)	R (%)	F (%)	P (%)	R (%)	F (%)
Comma	29.14	19.66	23.48	83.33	27.96	41.87
Period	86.46	90.64	88.50	86.31	97.52	91.58
Question mark	72.53	60.46	65.95	85.71	59.88	70.50
Exclamation mark	11.57	6.71	8.49	44.44	2.43	4.62
Micro-average	73.76	67.59	70.54	85.86	72.91	78.86

Table 4.3. Comparison of the CRF trained using all the features (CRF_3) and the CRF trained using the position and orthographic features (CRF_2.)

Predicting exclamation marks.

Correct sentence: đây là show của chúng ta!
Predicted sentence: đây là show của chúng ta.
(This is our show!)

Figure 4.1. An example of the model on predicting the wrong exclamation mark.

Figure 4.1 shows an example where the model predicts a period at the end of a sentence while the correct punctuation should be an exclamation mark. From our perspective, this is mainly because it failed in capturing conversation context or the speaker's attitude. To be more specific, since the model only looks at the current sentence to predict the punctuation, information from neighbor sentences is not considered. For example, when people are in an argument, they may want to express their opinions or show emotions resulting in more sentences ending with exclamation marks. Identifying the context or the speaker's attitude is rarely done sufficiently if we only look at one sentence each time. This requires the model to take into account information from the neighbor sentences or equivalently, model the long range dependencies over the observations, which is outside the scope of the linear-chain CRF model.

Predicting commas

Missing commas is also one case that hurts the performance. An example for this case is shown in Figure 4.2.

> **Correct sentence:** tôi thấy tiềm năng ở cậu, thám tử.
> (I see the potential in you, detective.)
> **Predicted sentence:** tôi thấy tiềm năng ở cậu thám tử.

Figure 4.2. An example of the models on missing the comma.

We conjecture that the first order Markov assumption of the linear-chain CRFs is the cause for this case. When words and corresponding labels around the current position can have an impact on the current punctuation, omitting such dependencies may lead to missing in predicting commas.

Also, there are cases where the use of commas which is not unique. An example is shown in Figure 4.3.

> **Correct sentence:** ờ thì, tôi phải công nhận hiện tại chưa có gì mới.
> **Predicted sentence:** ờ, thì tôi phải công nhận hiện tại chưa có gì mới.
> (Both sentences can be translated into: Umm, I have to admit that there is nothing new.)

Figure 4.3. An example of the usage of commas is not unique.

In this case, the model predicts the correct output in the sense that it is accepted by human. Sentences like the one in Figure 4.3 are frequently encountered during daily speeches and this proposes a challenge in standardizing the corpus and evaluating the performance of punctuation prediction systems.

Chapter 5

Conclusion

The previous chapters have introduced the problem of punctuation prediction in Vietnamese language. Throughout this thesis, we have discussed probabilistic graphical models and the conditional random field approach, one of the formulations of Markov Random Field. In chapter 3, we have illustrated an approach by modelling a punctuation prediction system as a sequence labelling problem and investigated how to apply conditional random fields efficiently. As discussed in Chapter 4, we have described several experiments and evaluate the corresponding performance.

In this chapter, we will conclude our current results and give further improvements for future works.

5.1 Future Works

Regarding the task of punctuation prediction, there are several aspects for exploration. In this work, we only exploit the word's identity at various positions in the sentence as main features and therefore, the set of feature templates is limited and not diverse. Other useful features such as part of speech of a word or gazetteer features, features from available dictionaries, might not be used in a straightforward way. The main reason is in order to correctly classify the part of speech for one word or to look up a word from a given dictionary, we need to have information about the capitalisation as well as the punctuations in a sentence [Nguyen et al., 2006, Tran et al., 2009]. However, since we are predicting the punctuations, this information

may be not available. In fact, our results from Table 4.2 show that using part of speech features without any caution worsens the performance. Incorporating other features for the punctuation prediction problem is not a trivial task and investigating on how to adapt these features presents a promising direction.

With respect to Vietnamese language processing, we take into account the way Vietnamese words are constructed. Different from English, Vietnamese words are made up from syllables and each syllable is separated by a white space [Dien et al., 2001, Nguyen et al., 2006]. Furthermore, each syllable itself has an individual meaning but when combined together, they may form a completely new word. Again, the main problem that we may not segment a sentence into a set of words is the existence of the missing information of capitalisation and punctuations. This property suggests that correctly segmenting a sentence as a preprocessing step may improve the performance of the punctuation prediction systems.

Finally, as we analyzed the errors in Section 4.4, building a standard corpus for this task or developing a model that can capture long range dependencies is one of the challenges.

5.2 Summary

This thesis has presented an approach to automatically assign punctuations to Vietnamese sentences. We have studied the linear-chain CRF and its applications in predicting the punctuations for the Vietnamese language. We have conducted experiments on a data set built from subtitles and this data set consists of many sentences closely related to daily conversation.

We have also presented various potential directions for further improving our system. The exploration of these directions can increase the performance of the system. Finally, the presented model - linear-chain conditional random field - has been specially designed for the task of sequence labelling and can be easily adapted to other related problems.

Bibliography

Cynthia Beath, Irma Becerra-Fernandez, Jeanne Ross, and James Short. Finding value in the information explosion. *MIT Sloan Management Review*, 53(4):18, 2012.

Doug Beeferman, Adam Berger, and John Lafferty. Cyberpunc: A lightweight punctuation annotation system for speech. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 1998.

D. P. Bertsekas. *Nonlinear Programming.* Athena Scientific, Belmont, MA, 1999.

Eunah Cho, Jan Niehues, and Alex Waibel. Segmentation and punctuation prediction in speech language translation using a monolingual translation system. In *Workshop on Spoken Language Translation*, 2012.

Nguyen Viet Cuong, Nan Ye, Wee Sun Lee, and Hai Leong Chieu. Conditional random field with high-order dependencies for sequence labeling and segmentation. *Journal of Machine Learning Research*, 15:981–1009, 2014.

Manning D. Christopher, Raghavan Prabhakar, and Schacetzel Hinrich. *An Introduction to Information Retrieval.* Cambridge, UK: Cambridge University Press, 2008.

Dinh Dien, Hoang Kiem, and Nguyen Van Toan. Vietnamese word segmentation. In *Natural Language Processing Pacific Rim Symposium*, 2001.

Ralph Grishman and Beth Sundheim. Message Understanding Conference-6: A Brief History. In *International Conference on Computational Linguistics (COLING)*, volume 96, pages 466–471, 1996.

Jing Huang and Geoffrey Zweig. Maximum entropy model for punctuation annotation from speech. In *International Conference on Spoken Language Processing*, 2002.

Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

John Lafferty, Andrew McCallum, and Fernando C.N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, 2001.

Dong C. Liu, Jorge Nocedal, Dong C. Liu, and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989.

Wei Lu and Hwee Tou Ng. Better punctuation prediction with dynamic conditional random fields. In *Conference on Empirical Methods in Natural Language Processing*, 2010.

Andrew McCallum, Dayne Freitag, and Fernando CN Pereira. Maximum entropy markov models for information extraction and segmentation. In *ICML*, volume 17, pages 591–598, 2000.

Ryan McDonald and Fernando Pereira. Identifying gene and protein mentions in text using conditional random fields. *BMC bioinformatics*, 6(Suppl 1):S6, 2005.

Cam-Tu Nguyen and Xuan-Hieu Phan. Jvnsegmenter: A java-based vietnamese word segmentation tool. *Online: http://jvnsegmenter. sourceforge. net*, 2007.

Cam-Tu Nguyen, Xuan-Hieu Phan, and Thu-Trang Nguyen. Jvntextpro: a tool to process vietnamese texts, 2005.

Cam-Tu Nguyen, Trung-Kien Nguyen, Xuan-Hieu Phan, Le-Minh Nguyen, and Quang-Thuy Ha. Vietnamese word segmentation with CRFs and SVMs: An investigation. In *Pacific Asia Conference on Language, Information and Computation*, 2006.

Michael Paul. Overview of the IWSLT 2009 evaluation campaign. In *Workshop on Spoken Language Translation*, 2009.

Lawrence Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989a.

Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *PROCEEDINGS OF THE IEEE*, pages 257–286, 1989b.

Charles Sutton and Andrew McCallum. An introduction to conditional random fields for relational learning. *Introduction to statistical relational learning*, pages 93–128, 2006.

Oanh Thi Tran, Cuong Anh Le, Thuy Quang Ha, and Quynh Hoang Le. An experimental study on Vietnamese POS tagging. In *International Conference on Asian Language Processing*, 2009.

Yanqing Zhao, Chaoyue Wang, and Guohong Fu. A CRF sequence labeling approach to Chinese punctuation prediction. In *Pacific Asia Conference on Language, Information and Computation*, 2012.