Fundamentals of Natural Language Processing

Negation and Uncertainty Detection using Classical and Machine Learning Techniques

Author One[a,1]**, Author Two**[b,2]**, Author Three**[c,3] **and Author Four**[d,4]

[a]*NIU of Author one*
[b]*NIU of author two*
[c]*NIU of author three*
[d]*NIU of author four*

Abstract—An abstract is a brief summary that outlines the key aspects of a work. An example of a famous abstract is reproduced verbatim here for illustration purposes [1]: The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 Englishto-German translation task, improving over the existing best results.

Keywords—*a, b, c, d*

Contents

1. Introduction

This sample document contains indications about how to write the report for the project in the subject of Natural Language Processing. The full report must be no longer than 5 pages without including references. **Any additional pages will not be evaluated**. The section structure is open, but you are encouraged to follow a principled academic writing style (you may mimic that of the papers provided as references). **Form will be taken into account in the evaluation of the project.**

2. Writing LATEX

LATEXis the typesetting tool that most academics use these days. For this reason, you are required to present your work using it. LATEXhas a bit of a steep learning curve at the beginning, but you will see that it is quite intuitive and you will get progressively more productive at it as you write.

2.1. Basics

LATEXis based on the idea of defining commands (or macros) that control the presentation of the text on a document. As a matter of fact, LATEXignores whitespace except for double newlines (pressing intro twice), which is used to delimit paragraphs. Commands are usually preceeded by a backslash ( ) and have a number of arguments, if any, between braces ({ and }). You can set a **bold** or an *italic* typeface using the `\textbf{.}` and `\textit{.}` commands. You can also change the size of the font for a scope. You can change text color using the `\color{col}` command.

Nevertheless, you should mostly refrain from formatting text explicitly. LATEXprovides commands for recurring types of text modifications with some batteries included. For instance, if you want to divide your text in chunks under prominent titles you can use the `\section{.}`, `\subsection{.}`, `\subsubsection{.}` ... etc commands, which will handle everything for you. These titles will automatically appear in the table of contents, they will be properly numbered in the order of appearance, etc. If you are writing a book, you can also use the `\part{.}` and `\chapter{.}` commands for this.

Many LATEXconstructs are built around the `\begin{environment}` and `\end{environment}` commands. You write some text between these two commands that is to be formatted according to the rules of the given environment. With this, you can write

- Bullet point lists (environment itemize)

1. Enumerations (environment enumerate)

> **Information**
>
> ... or some custom stuff as well. The possibilities are quite endless, really.

You can edit LATEXin any text editor of your choice. I suggest that you try Overleaf first, because the editing experience is very simple and very close to that of a shared Google Docs. These days they have gotten quite stingy however, so I suggest using VS Code with the LATEXWorkshop extension instead. If you need to share the document, just put it in a Git repository. There are a myriad alternatives, so go out there and try what works for you.
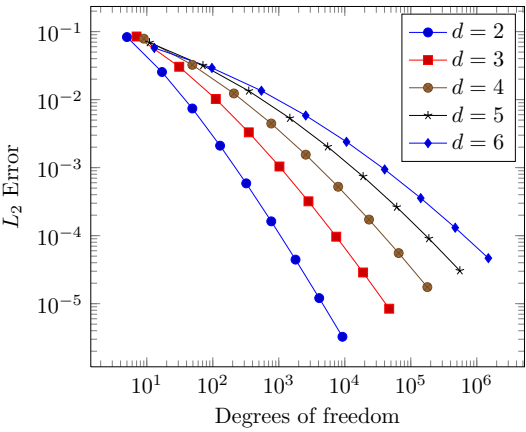
3. Figures and tables

3.1. Figures

Fig. 1 shows an example figure. Figures can be referenced using `\ref{ident}`, where name is a previously-defined `\label{name}` for any object in the document. You can actually also do this with sections (Section 3) or equations or other types of objects that will be seen below.
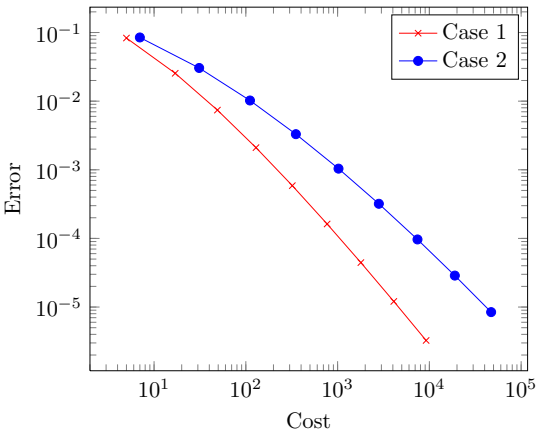
(a) Example left figure.

(b) Example right figure.

Figure 2. Example figure that covers the width of the page obtained from PGFPlots [2].
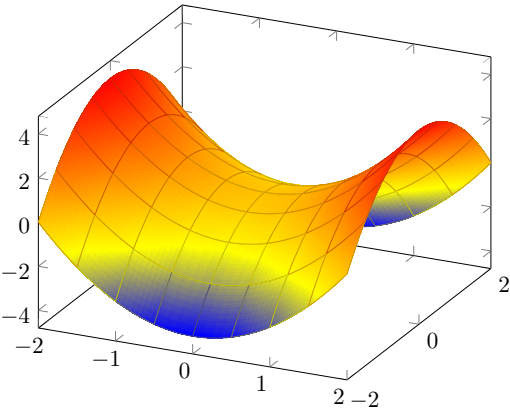

Figure 1. Example figure obtained from PGFPlots [2].

Fig. 2 shows an example of two figures that cover the width of the page. It can be placed at the top or bottom of the page. The space between the figures can also be changed using the `\hspace{Xpt}` command.

> **Information**
>
> Figures in LATEXare known to be quite anarchic. You should always wait until the very end to place them well into the document. In any case, you should always reference the Figure and provide a Caption so that it can be contextualised even when it lies far away from the text that accompanies it.

> **Information**
>
> Avoid raster graphics (jpeg, png) for figures if you can. Always use vectorial formats for plots and diagrams. The preferred formats for vectorial images are SVGs converted to Encapsulated Post-Script (.eps, you can use Inkscape to perform this conversion) or PDF files. Regardless of the image format you are using, generate reasonably high-resolution files (the usual is 300dpi, which means 300 pixels per squared inch). **Keep in mind that Matplotlib can be configured to generate plots with these settings.**

3.2. Tables

Table 1 shows an example table. The `\tabletext{}` is used to add notes to tables easily.

> **Information**
>
> This template uses the `booktabs` package by default. This package makes typesetting nice tables easier, but it imposes some general design guidelines. The most prominent is the distinction between middle and edge rules (horizontal lines) and the discouragement of vertical lines.

> **Information**
>
> You should **always** avoid typesetting tables by hand. Use tools such as this one to spare yourself some unnecessary pain.

Table 1. Small example table.

Column 1	Column 2
Data 1	Data 2
Data 3	Data 4

Note: I'm a table text for additional information.

4. Tau packages

4.1. Tauenvs

This template has its own environment package *tauenvs.sty* designed to enhance the presentation of the document. Among these custom environments are *tauenv*, *info* and *note*.

There are two environments which have a predefined title. These can be included by the command `\begin{note}` and `\begin{info}`. All the environments have the same style.

An example using the tau environment is shown below.

> **Environment with custom title**
>
> This is an example of the custom title environment. To add a title type `[frametitle=Your title]` next to the beginning of the environment (as shown in this example).

Tauenv is the only environment that you can customize its title. On the other hand, info and note adapt their title to Spanish automatically when this language package is defined.

4.2. Taubabel

In this new version, we have included a package called *taubabel*, which have all the commands that automatically translate from English to Spanish when this language package is defined.

By default, tau displays its content in English. However, at the beginning of the document you will find a recommendation when writing in Spanish.

Note: You may modify this package if you want to use other language than English or Spanish. This will make easier to translate the document without having to modify the class document.

5. Equation

Equation 1, shows the Schrödinger equation as an example.

$$\frac{\hbar^2}{2m}\nabla^2\Psi + V(\mathbf{r})\Psi = -i\hbar\frac{\partial\Psi}{\partial t} \tag{1}$$

The *amssymb* package was not necessary to include, because stix2 font incorporates mathematical symbols for writing quality equations. In case you choose another font, uncomment this package in tau-class/tau.cls/math packages.

If you want to change the values that adjust the spacing above and below the equations, play with `\setlength{\eqskip}{8pt}` value until the preferred spacing is set.

6. Adding codes

This class[1] includes the *listings* package, which offers customized features for adding codes in LaTeX documents specifically for C, C++, LaTeX and Matlab.

You can customize the format in tau-class/tau.cls/listings style.

```
function fibonacci_sequence(num_terms)
    % Initialize the first two terms of the
    sequence
    fib_sequence = [0, 1];

    if num_terms < 1
        disp('Number of terms should be greater
    than or equal to 1.');
        return;
    elseif num_terms == 1
        fprintf('Fibonacci Sequence:\n%d\n',
    fib_sequence(1));
        return;
    elseif num_terms == 2
        fprintf('Fibonacci Sequence:\n%d\n%d\n',
     fib_sequence(1), fib_sequence(2));
        return;
    end

    % Calculate and display the Fibonacci
    sequence
    for i = 3:num_terms
        fib_sequence(i) = fib_sequence(i-1) +
    fib_sequence(i-2);
    end

    fprintf('Fibonacci Sequence:\n');
    disp(fib_sequence);
end
```

Code 1. Example of Matlab code.

If line numbering is defined at the beginning of the document, I recommend placing the command `\nolinenumbers` at the start and `\linenumbers` at the end of the code.

This will temporarily remove line numbering and the code will look better as shown in Code 1.

7. References

The default formatting for references follows the IEEE style. You can modify the style of your references, for that, go to tau-class/tau.cls/biblatex. See appendix for more information.

---
[1] Hello there! I am a footnote :)

8. Appendix

8.1. Alternative title

You can make the following modification in tau-class/tau.cls/title preferences section to change the position of the title.

```
\newcommand{\titlepos}{\centering}
```

Code 2. Alternative title.

This will move the title to the center.

8.2. Info environment

An example of the info environment declared in the tauenvs.sty package is shown below. Remember that *info* and *note* are the only packages that translate their title (English or Spanish).

> **Information**
> Small example of info environment.

8.3. Equation skip value

With the `\eqskip` command you can change the spacing for equations. The default *eqskip* value is 8pt.

```
\newlength{\eqskip}\setlength{\eqskip}{8pt}
  \expandafter\def\expandafter\normalsize\
    expandafter{%
    \normalsize%
    \setlength\abovedisplayskip{\eqskip}%
    \setlength\belowdisplayskip{\eqskip}%
    \setlength\abovedisplayshortskip{\eqskip-\
    baselineskip}%
    \setlength\belowdisplayshortskip{\eqskip}%
  }
```

Code 3. Equation skip code.

8.4. References

In case you require another reference style, you can go to tau-class/tau.cls/biblatex and modify the following.

```
\RequirePackage[
   backend=biber,
   style=ieee,
   sorting=ynt
]{biblatex}
```

Code 4. References style.

By default, *tau class* has its own .bib for this example, if you want to name your own bib file, change the *addbibresource*.

```
\addbibresource{tau.bib}
```

9. Acknowledgements

Tau LaTeX template built by Guillermo Jimenez.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, *Attention Is All You Need*, arXiv:1706.03762 [cs] version: 5, Dec. 2017. DOI: 10.48550/arXiv.1706.03762. [Online]. Available: http://arxiv.org/abs/1706.03762 (visited on 10/25/2022).

[2] *PGFPlots - A LaTeX package to create plots*. [Online]. Available: https://pgfplots.sourceforge.net/.