1. 使用不同大小的窗口，来观察校验集和测试集的表现，并给出分析得出结论；
2. 使用不同的学习率；优化器；
3. 使用不同的隐含层数和不同维度的隐含层；
4. 使用不同的激活函数；
5. 不掉包来实现一个RNN网络；
6. 使用梯度裁剪技术；
7. 使用不同的embedding size;

%

% File acl2021.tex

%

%% Based on the style files for EMNLP 2020, which were

%% Based on the style files for ACL 2020, which were

%% Based on the style files for ACL 2018, NAACL 2018/19, which were

%% Based on the style files for ACL-2015, with some improvements

%% taken from the NAACL-2016 style

%% Based on the style files for ACL-2014, which were, in turn,

%% based on ACL-2013, ACL-2012, ACL-2011, ACL-2010, ACL-IJCNLP-2009,

%% EACL-2009, IJCNLP-2008...

%% Based on the style files for EACL 2006 by

%%e.agirre@ehu.es or Sergi.Balari@uab.es

%% and that of ACL 08 by Joakim Nivre and Noah Smith

\documentclass[11pt,a4paper]{article}

\usepackage[hyperref]{acl2021}

\usepackage{times}

\usepackage{latexsym}

\renewcommand{\UrlFont}{\ttfamily\small}

% This is not strictly necessary, and may be commented out,

% but it will improve the layout of the manuscript,

% and will typically save some space.

\usepackage{microtype}

%\aclfinalcopy % Uncomment this line for the final submission

%\def\aclpaperid{\*\*\*} % Enter the acl Paper ID here

%\setlength\titlebox{5cm}

% You can expand the titlebox if you need extra space

% to show all the authors. Please do not make the titlebox

% smaller than 5cm (the original size); we will check this

% in the camera-ready version and ask you to change it back.

\newcommand\BibTeX{B\textsc{ib}\TeX}

\title{Works based on NNLM}

\author{

\textbf{Pingwei Sun} \\

\\

NEU-NLP Lab \\

}

\date{}

\begin{document}

\maketitle

\begin{abstract}

During the second stage of our course, I have done works based on Neural Network Language Model.

Both a FFN and a RNN are built up.

As for further research, I have done some related works based on those models, which includes

experiments on hyper-parameters, comparing performance of various models and introducing pre-trained layers.

The details of works are presented as follow.

\end{abstract}

\section{Introduction}

With the improvement of computer processing ability, people have been exploring the method of processing natural language by programs.

One of the most important tasks is to create a language model, which can effectively generate mathematical model according to our natural language.

Natural language model has gone through the stages of grammatical model, statistical model and now stepped into the neural network stage.

Different from the previous dictionary based theory, \textbf{NNLM}\citep{bengio2000neural} will transform natural language into a set of machine's knowledge system by learning from data.

There are no artificial assumptions during the whole process and will also save a lot of storage cost.

However this work did not take the sequential information of language into consideration.

To make models able to form memory, \textbf{RNN}\citep{mikolov2010recurrent} was launched representing history by neurons

with recurrent connections.

\textbf{Based on NNLM, I have conducted the following experiments:}

\begin{itemize}

\item [1)] Explore the impact of hyper-parameters(step, embedded size, hidden size) on the performance of the model.

\item [2)] Compare the performance of different models.

\item [3)] Introduce pretrained embedding parameters into the modeel.

\end{itemize}

\section{Methods}

\subsection{Vocabulary Construction}

Here I customize the method of constructing the vocabulary by myself.

First, document is opened by the program. Then the words are separated by space(ways of splitting can be changed by calling various APIs), and stored in a \emph{frequency dict} in the pattern of \emph{\{word:frequency\}}.

Then we need to build two \emph{dicts}, \emph{word2vecotr and vector2word}, to generate a dual-direct index. Special words such as \emph{[pad], [unk], [sos], [eos]} come first, and dual-direct index pairs are built according to the \emph{frequency dict}.

\subsection{Dataset Construction}

To feed the model with appropriate data, I rewrite the \emph{Dataset} method.

Specifically, input corpus are translated into numbers according to the previous dual index sentence by sentence.

Then, \emph{[[n-input], [target]} corpus list are generated from sentences, following the instruction of \emph{n-step}, and those being unqualified in length will be padded in front.

With the customized collate-fn, all records in the corpus list are packaged as tuples, gained by \emph{Dataloader} and finally make up batches.

\subsection{Pretrained embedding}

In my second and third works, it is necessary to load part of parameters of pretrained models.

To implement this function, we need to iterates through the \emph{state\\_dict} of a model and load the necessary items by checking their keys.

\subsection{Facilitation for experiments}

To make it more convenient to conduct my test, I use \emph{argparse API} to change and save settings of hyper-parameters. In addition, \emph{tensorboard} is called to visualize the training process.

\section{Experiments}

\label{sec:length}

Workshop chairs may have different rules for allowed length and whether supplemental material is welcome.

As always, the respective call for papers is the authoritative source.

\section{Conclusion}

During the second stage of the course, I build up various neural network models and conducted experiments to figure out relations between certain hyper-parameters and model performance.

In addition, I compare models by their performance and try some popular tricks, such as pretraining, to make them work better.

\paragraph{}

Finally, I would like to give thanks to our tutors and seniors.

I have learnt a lot and had a pretty good time in the last week!

\bibliographystyle{acl\_natbib}

\bibliography{mycite}

%\appendix

\end{document}