Seismic Phase Picking

Junqing Chen
Yuanpei College
Peking University
2000017795@stu.pku.edu.cn

Tianxing Fan
School of Computer Science
Peking University
carlosfan2001@outlook.com

Abstract

Seismic Phase Picking aims to predict the P-wave and S-wave arrival time from the wave images. It has a far-reaching effect on monitoring geological activity. However, traditionally, these images are manually annotated, which is time consuming and error-prone. In the past century, statistic methods have been proposed but their accuracy is far below human level. In this article, we explore to apply neural networks on the task and compare its effects with traditional methods.

1 Introduction

After an earthquake, it is an urgent need to compute the location, magnitude and other attributes of the earthquake, the very first step of which is seismic phase picking. Seismic phase picking means predicting the P-wave and S-wave arrival time from the collected waveforms. It had to be manually annotated for a very long time, which is laborious and error-prone, as the human analyst has to observe waveforms of a very long time and precisely pick only two timestamps. In the 1990s, statistic methods are proposed successively. However, the accuracy of statistic methods is very low. Comparing to P-wave, the accuracy of statistic methods on the S-wave arrival time prediction is a little worse, which has to do with the geological fact that P-wave comes first and may influence the collected S-wave waveform. In other words, the statistic methods are pretty sensitive to the background noise, which in reality is unavoidable.

Therefore, in this article, we try to apply the deep neural networks on the seismic phase picking task. We hope that the huge amount of learnable parameters in the neural network can make it robust to the backgroumd noise and have a better performance than statistic methods.

2 Related Work

In this section, we mainly focus on the related works which apply deep neural networks to the task. Before RNN is widely used, Zhu et al. [4] puts forward an architecture composed of convolutional layers and fully-connected layers. In their work, they use the convolutional layers to expand the original 3-channel waveforms(E, N, Z) to 64 channels. And pooling layers are employed to make the waveform with an original length 2000 shortened to length 1. So that the final vector with a length 64 can be used as input to the fully-connected layer and get predictions about P-phase and S-phase arrival time.

Later, as the task has a lot to do with temporal dependency and RNN is powerful in processing data with timing information, architectures including RNN and its variants are proposed successively. Zhou et al. [3] uses GRUs to grasp the temporal dependency in the waveform and dramatically improves the network's performance. Then comes the attention mechanism. SM et al. [2] explores to combine attention mechanism, LSTMs and transformers all together, which gets an even better performance. However, there is a few difference in employing timing information in these two architectures. Zhou et al. [3] slices the time interval and transform every time interval to a vector as input to the GRU. SM et al. [2] seems to learn from the very first work Zhu et al. [4]. It uses convolutional layers to expand the E, N, Z channels to 64 channels and takes the 64-dimentional vector as input vector to the

LSTM. It is even an interesting research topic to compare which way is better at grasping temporal dependency.

3 Traditional method

The adopted traditional method is AR picker Akazawa [1]. There are 8 hyperparameters to finetune, and the time cost for finetuning goes exponentially as the number of hyperparameters increases. For the sake of saving time, we only finetune two hyperparameters, f1 and f2, while the other six are set to default. f1 determines the frequency of the lower bandpass window and f2 determines the frequency of the upper bandpass window. We test 4 different values each and complete 16 experiments. For the experiment, we only use 200 samples. Fig. 1 shows the results for different f1 and f2 values.

In comparision, the default f1 and f2 values get mean P-wave arrival time 3.57695 and mean S-wave arrival time 14.71305. Therefore, we thought f1=1.2 and f2=18.0, or f1=1.5 and f2=22.0 are two good choices. We test on these two settings with 1000 samples. And Fig. 2 shows the final results. To give a clear illustration, we only show the distribution of residuals less than 1s. The x_label for each image shows how many predictions get a residual less than 1s out of all the 1000 predictions.

In conclusion, the traditional method sometimes give a reasonably good result. But its variance is really large and many predictions simply have nothing to do with the correct arrival time. Therefore, it is hard to completely substitute traditional methods for manual annotation.

4 Deep Neural Network

Generally speaking, the key to solve seismic phase picking is to employ the time dependency in the waveform, which is where RNN, LSTM and more powerful transformers stand out. Therefore, one intuition to design the neural network architechture is to use transformers to analyze the time dependencies in the waveform.

However, there are still two main problems to solve. One is to construct the input to the transformer. As transformer was originally designed to solve NLP problems, it could not directly take an image as input. To construct an appropriate input, we have to use deep convolutional neural networks to abstract the information in the waveform image and the information as input to the transformer. The other problem is to get the prediction from the transformer's output. Transformers output a vector for every timestamp so we can just take the output vector and use a feed forward network to predict the probability that P-wave arrives at this timestamp. Then we take a argmax over the probabilities and timestamp with the highest probability is the predicted P-wave arrival time or S-wave arrival time.

Above are just our general ideas. In practice, it is extremely hard to construct a huge and efficient network from pieces. For reference, according to the SM et al. [2], they train the network on 4 parallel Tesla V-100 GPUs for 89 hours. So we simply reference their work and directly apply their model on our dataset. However, due to their model is designed to process waveform with a length of 6000, we have to truncate our waveform first so that the model works. Fig. 3 shows the final result. For those sample whose P-wave arrival sample or S-wave arrival sample is larger than 6000th sample, we simply ignore it, for which the shown total samples in the bottom of the picture is 9986. To our surprise, the model works badly and even worse than traditional method. We think the main reason is that our data format is quite different from the original training dataset so a finetuning needs to be done first.

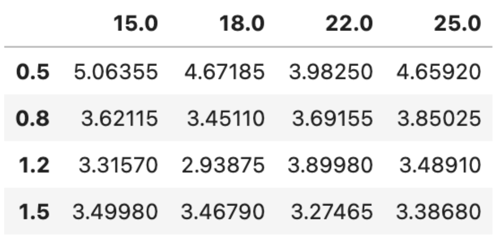
5 Evaluation

In last two sections, we simply show the residual distribution of ARpicker and EQtransformer. In this section, we will rigorously evaluate the two methods and compute some mathematical metrics to show their performances.

Due to the extremely low computing speed of ARpicker, about 1 iter/s, we only use 2000 samples for evaluation. It is noted that for EQtransformer we have to ignore samples whose P-arrival sample or S-arrival sample is larger than 6000 as the data has to be truncated to 6000 samples to meet the model's requirement for input length. The final result is shown in the Tab. 1 below. The residual distribution on the evaluation set is shown in Fig. 4

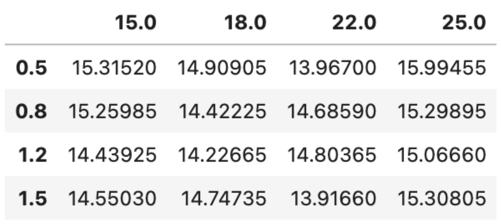
Table 1: Evaluation Results

	mean_error	var_error	precision	recall	f1_score
EQT_P_wave	12.42171	12.42171	0.00228	0.0015	0.001809
EQT_S_wave	11.28084	11.28084	0.005319	0.0035	0.004222
ARpicker_P_wave	3.52729	3.52729	0.336	0.336	0.336
ARpicker_S_wave	15.89543	15.89543	0.095	0.095	0.095



	15.0	18.0	22.0	25.0
0.5	5.06355	4.67185	3.98250	4.65920
0.8	3.62115	3.45110	3.69155	3.85025
1.2	3.31570	2.93875	3.89980	3.48910
1.5	3.49980	3.46790	3.27465	3.38680

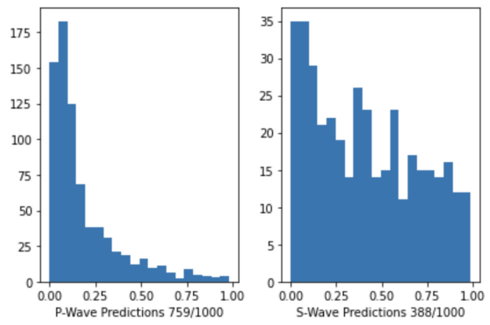
(a) Mean residuals for P-wave arrival time



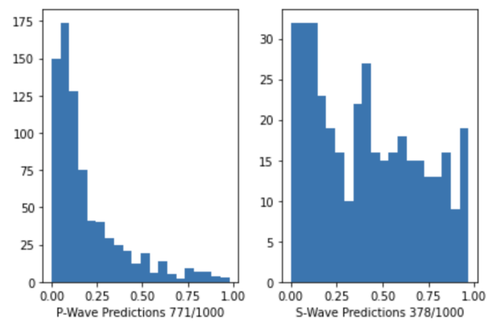
	15.0	18.0	22.0	25.0
0.5	15.31520	14.90905	13.96700	15.99455
0.8	15.25985	14.42225	14.68590	15.29895
1.2	14.43925	14.22665	14.80365	15.06660
1.5	14.55030	14.74735	13.91660	15.30805

(b) Mean residuals for S-wave arrival time

Figure 1: Row index show f1 values and column index show f2 values

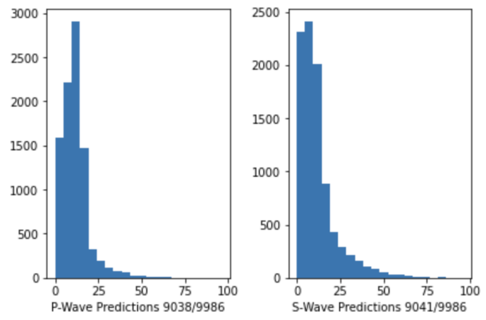


(a) f1=1.2, f2=18.0

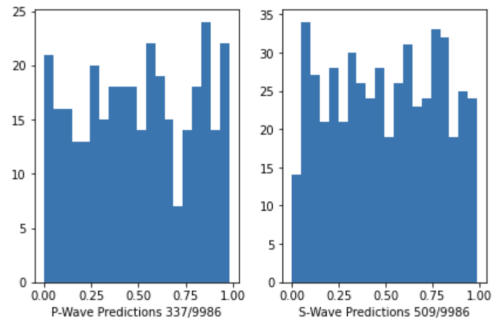


(b) f1=1.5, f2=22.0

Figure 2: Final results for traditional methods



(a) Results on whole 10000 samples



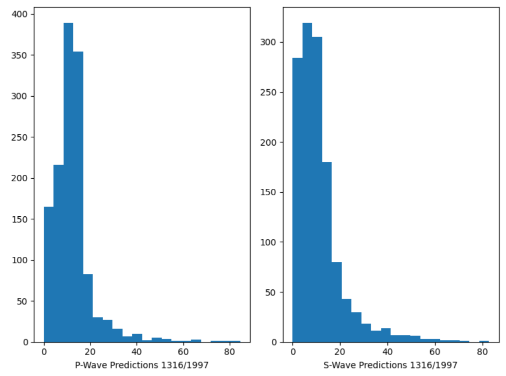
(b) Distribution of residuals less than 1s

Figure 3: Final results for EQTransformer

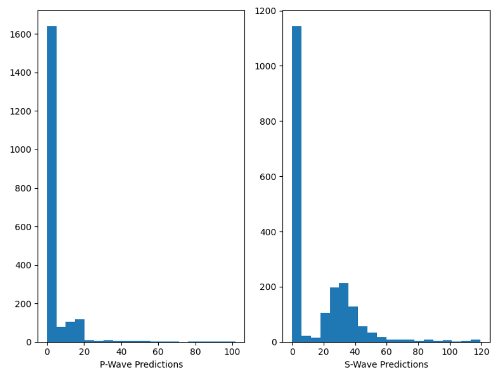
As talked in last section, because we directly use the model proposed in SM et al. [2] and the data formats in two datasets are quite different, it is no wonder that the model performs badly without a finetuning.

6 Conclusion

From our experiments, it is clear that traditional methods perform much better on P-phase picking than S-phase picking. ARpicker actually gets a mean error of about 3.5s for P-phase picking, while for S-phase picking, the mean error quickly increases to about 16s, which is even worse than the EQtranformer without finetuning.



(a) Residual distribution of EQtransformer



(b) Residual distribution of ARpicker

Figure 4: Residual distributions on the evaluation set

Our experiments also illustrate the inflexibility of neural network. For traditional methods, it doesn't matter how long the input sequence is. However, for neural networks, there must be a prior setting for the input waveform length as, for example, the inside fully-connected layer has a fixed dimension. Besides, according to the original paper SM et al. [2], the model has a great generalization and makes a good performance in Japan waveform data which is not included in the training dataset. According to our experiments, it seems that a finetuning is needed, though the bad results might be mainly caused by the different data formats.

Finally, I just want to say a sorry. We definitely realize how basic our work is. It just bothers us a lot to solve all kinds of errors when trying to finetune the EQtransformer on our dataset. Even when we just ran the example training program, it crashed. The terrible situation has a lot to do with the package dependency. The released EQtransformer package is for python 3.5-3.6. Our server has a python version 3.8, for which pip simply crashes and we have to install EQtransformer from the github source code. I spent days of days in modifying the original setup.py and finally figured out a set of package versions that works for directly applying the model on our dataset but doesn't work when trying to finetune the model on our dataset.

Anyway, if I had started earlier and worked harder for the project, I might have even been able to construct a network from pieces as I finally figured out how to apply LSTM, transformers on the waveform data several days ago. It used to confuse me how these papers manage to apply LSTM or transformer on the waveform data which only has 3 channels. It is definitely not suitable to use the 3-dimension vector as input to LSTM or transformer. It is after I read Zhou et al. [3] and saw output shape of every layer inside EQtransformer that I understand that the key is to use convolutional layers to expand the E,N,Z channels and take the expanded vector as input to LSTM or transformer.

7 Division of duty

Junqing Chen finishes all the coding and writing, Tianxing Fan has several discussions with Junqing Chen and provides some inspirations.

References

[1] Akazawa. Technique for automatic detection of onset time of p- and s-phases in strong motion records. *Vancouver*, 786:786, 2004. 2

[2] Mousavi SM, Ellsworth WL, and Zhu WQ. Earthquake transformer—an attentive deep-learning model for simultaneous earthquake detection and phase picking. *Nat. Commun.*, 11:3952, 2020. 1, 2, 3, 4

[3] Yijian Zhou, Han Yue, Qingkai Kong, and Shiyong Zhou. Hybrid event detection and phase-picking algorithm using convolutional and recurrent neural networks. *Seismological Research Letters*, 90(3):1079–1087, 2019. 1, 4

[4] Lijun Zhu, Zhigang Peng, James McClellan, Chenyu Li, Dongdong Yao, Zefeng Li, and Lihua Fang. Deep learning for seismic phase detection and picking in the aftershock zone of 2008 mw7. 9 wenchuan earthquake. *Physics of the Earth and Planetary Interiors*, 293:106261, 2019. 1