Deep Learning for Seismic Data Classification in Full-waveform Inversion

Congyue Cui
Department of Geosciences
ccui@princeton.edu

Chao Song
Department of Geosciences
chaosong@princeton.edu

Abstract

Full waveform inversion images the internal structure of the Earth by minimizing the difference between observed and synthetic seismograms. In order to improve the convergence, we must choose seismogram pairs that are similar enough. In this project, we aim to develop a new method to select more qualified seismograms that are suitable for full-waveform inversion. This is regarded as a multinomial classification problem, which gives a grade ranking of a seismogram (3 classes). Impressed by the outstanding performance of Deep Residual Network (ResNet) and applications of other convolutional neural networks in seismology, we choose the ResNet architecture as our first step to develop a minimum viable product. We manage to train the model based on data with labels determined by the time-frequency misfit and successfully predict the class labels of data pairs in test set with a acceptable accuracy. This preliminary model still has much space in both data preprocessing and architecture itself to be improved. We hope to continuously fine-tune this product or try out other architectures for improving its efficiency and accuracy, and promote it in the future.

1 Motivation

An *earthquake* represents the shaking of the surface of the Earth, resulting from the sudden energy release of fault slip underground the Earth that creates seismic waves. A *seismogram* is a wiggle-like graph output by a digital converter from these seismic waves at the seismic measuring station. It is a 3-component record of the ground motion as a function of time in the three orthogonal directions (see Figure 1 as an example). A typical seismogram is expected to record the energy from an earthquake, but not limited to that. Almost all shaking signals such as an explosion, construction, background noise can be captured by a seismogram. We use seismograms as our primary data source to investigate the *internal structure of the Earth*, since the high-pressure and high-temperature nature of the Earth's interior making it impossible to penetrate into deeper regions. The structure of the Earth is a long-lasting scientific goal inside both seismology and, more generally, Earth sciences community. In this field, we call this process from seismograms (data) to unknown structures, unknown sources or others (models) as *inversion*. To fully exploit the information and ensuring the full physics of seismic wave propagation, a process referred to as seismic Full-Waveform Inversion (**FWI**) which idea is first proposed in [1]. It is a optimization problem where full-wave equation modeling is performed at each iteration of the optimization in the final model of the previous iteration. In other words, it tries to iteratively minimize the difference between the observed and synthetic seismograms, where synthetics are generated by a forward simulation based on the structure model inverted from the previous iteration. Even though a lot of theoretical and experimental researches has been explored since then, FWI has not been recognized as an efficient seismic imaging techniques due to the massive computations required [2]. It becomes more and more popular as the development of high-performance computing resources like supercomputers and Graphics Processing Unit (GPU) acceleration. One the one hand, to obtain a global 3-dimensional structure of the Earth, we want to employ data as much as possible that has the best coverage over the world.

On the other hand, to ensure convergence, we have to only select those seismograms that are close enough to their corresponding synthetic one, which means that we are using a small portion of data although there are numerous data out there. The algorithm we have applied to ensure the data quality is reasonable, but seems too strict to use all useful information. In the meantime, some similarities between observations and synthetics that can be identified by human intervention could hardly be detected by our current algorithm. Therefore, our goal in this project is to prepare as much qualified data as possible for the consequent FWI. In the machine learning perspective, we need to build a classifier to predict whether a trace of seismogram is qualified (1) or not (0) for FWI. Out of the potential same nature as face recognition challenge, we are putting some faith on *neural network* in deep learning to tackle this problem.

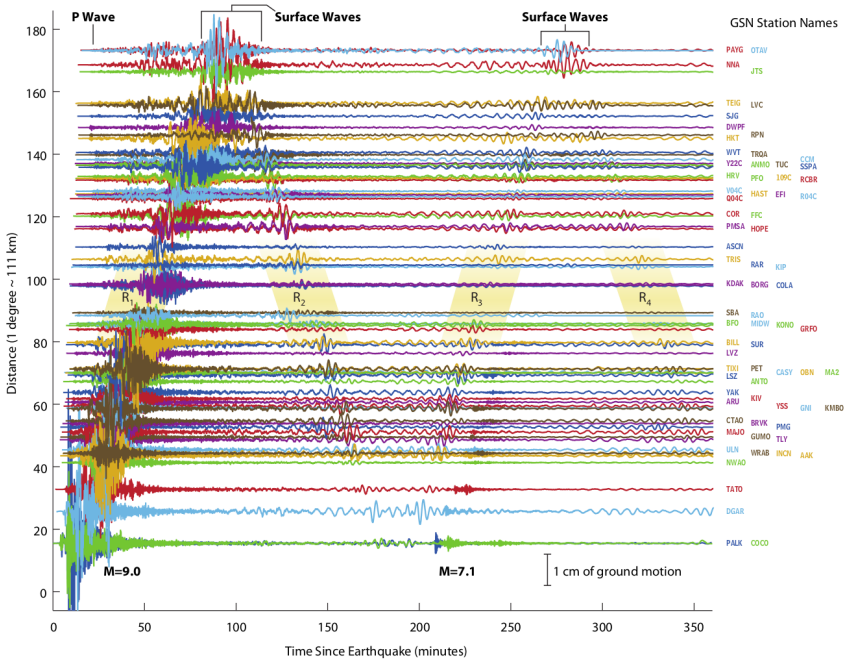

Figure 1: Seismograms recorded during 2004 Sumatra-Andaman Earthquake, modified from [3]

2 Related Work

In the Introduction section, we mentioned that the first original idea about FWI was actually proposed by Albert Tarantola in 1984 [1], though this word had not been invented yet. All types of waves are involved in this optimization, including body waves with diverse ray paths (e.g. direct waves, multi-reflected waves, waves diving deeply into the core) and surface waves. Body wave is the type that travel through the interior of the Earth and are controlled by the material properties. It can be divided into compressional wave that displaces the ground in the same direction as its propagation and shear wave that displace the ground perpendicular to its propagation. Surface wave is the type that travel along the ground surface due to its rapid attenuation downward away from the surface. A lot of previous work has contributed to the development of theory and application of FWI (e.g. [4], [5], [6], [7], [8], [9], [10]). In our research, we are using a GPU-accerated version the 3D spectral-element FWI solver SPECFEM3D_GLOBE ([11], [12], [13], [7]). However, high frequency surface waves are giving us a lot of trouble in the process of selecting qualified data for inversion. So we want to low down the frequency and use more data. Although some of the similarities between surface waves can be identified by human, they can hardly be detected by our current algorithm.

In this assignment, we are going to use *deep residual network* (ResNet), one kind of convolutional neural network (CNN). CNN belongs to a broader class of deep neural networks commonly applied to analyzing visual imagery. It were inspired by the biological processes that the connectivity between neurons resembles the organization of the animal visual cortex ([14]). As a classification

algorithm, CNN is basically designed to contain the input/output layer, several hidden blocks involving a convolutional layer, Rectified Linear Unit (ReLU, employing activation function defined as the positive part of its argument) layer, pooling layer (downsample the spatial size of the representation to reduce parameters), fully-connected layer (full connection to all activations in the previous layer) and normalization layer (simulation of inhibition schemes observed in the biological brain). Inspired by the great idea by Kunihito Fukushima [14], the first convolutional network *time delay neural network* (TDNN) [15] and its following variants (e.g. [16], [17]) was generated. In the 2000s, CNN were starting to get breakthrough by the implementations on GPUs ([18], [19], [20], [21], [22], [23]). The latest state-of-art CNN before ResNet was the 22-layer-deep GoogleLeNet [23], who won the 1st place in ImageNet Large-Scale Visual Recognition Challenge 2014 (ILSVRC14). In the context of our project, we believe that image analysis (classification specifically) is similar to the time series analysis, only from 2-D to 1-D. The difference from ordinary neural networks that CNN makes the explicit assumption that the inputs are images allows us to encode certain properties into the architecture [24].

Based on the remarkable achievements of previous CNNs, Kaiming He et al. (2016) [25] blew people's mind in 2015 with his most groundbreaking work, ResNet, which solved the degradation problem and thus made training up to hundreds or even a thousand layer possible [26], compared to its ancestors, AlexNet (5 convolutional layers [21]), VGG network (19 convolutional layers [22]) and GoogleLeNet (22 convolutional layers [23]). Due to its state-of-art performance, several variants emerged based on its architecture (e.g. [27], [28], [29], [30]) and are undoubtedly making this network much stronger.

The challenge we want to solve here belongs to a broader concept called *Time Series Classification* (TSC) [31]. Since CNNs have successful applications in many domains like natural language processing besides image recognition, it also starts to motivate researchers to adopt them for time series analysis (e.g. [32], [33], [34]). And CNNs, with architecture informed by the inverse problem itself, have already had applications in the seismology analysis (e.g. [35], [36], [37]), given the overview of meachine learning in solid Earth geosciences [38]. Out of the results from comparisons of 8,730 deep learning models on 97 time series datasets in [31], Resnet stands out to show the best comprehensive performance among all, so we decide to choose it to tackle our seismogram classification in the end.

3 Data Description

3.1 Source and Tools

We are using `ObspyDMT`, a Python toolbox [39], to retrieve, process and manage our seismic datasets [1]. The facilities of IRIS (Incorporated Research Institutions for Seismology) Data Services, and specifically the IRIS Data Management Center, were used for access to all seismograms and related metadata used in this study. IRIS Data Services are funded through the Seismological Facilities for the Advancement of Geoscience and EarthScope (SAGE) Proposal of the National Science Foundation under Cooperative Agreement EAR-1261681 [2].

3.2 Data-preprocessing

Our complete raw data include 27,489 pairs of synthetic and observed seismograms. Not all of them will join the training, and we will discuss it in detail later. The number of seismogram pairs that are put to the training set will increase over time. We only use the Z component. All data is cut to be 30-min long and then low-pass filtered below than 50 s. To decrease the data size, it is then down-sampled from 18,000 to 1024 samples for each trace.

We use time-frequency misfits to quantitatively assess the similarity between synthetic and observed seismograms ([40], [41]), which means the data is processed in time-frequency domain. This algorithm is chosen because of the following advantages.

1. Complete quantification of seismic waveform misfit in the frequency range of interest.

[1] https://github.com/kasra-hosseini/obspyDMT

[2] http://ds.iris.edu/ds/

2. Separation of phase and amplitude information.
3. Relaxation of the requirements on waveform similarity needed for the measurement of pure cross-correlation time shifts.
4. Possibility to analyse complete seismograms including body waves, surface waves and interfering phases.

We denote the i-component of an observed seismogram at position $\mathbf{x} = \mathbf{x}^r$ as $u_i^0(x^r, t)$. The synthetic is denoted as $u_i(\mathbf{m}; \mathbf{x}^r, t)$, where $\mathbf{m}$ is the Earth model. For brevity, we omit $\mathbf{x}^r$ and $\mathbf{m}$. Now we can analyze how the frequency content of the data evolves with time by calculating the Fourier transform of $u_i^0(t)$ multiplied by a sliding window function $h(t-\tau)$ centred around τ [9]:

$$\widetilde{u}_i^0(t,\omega) = F_h[u_i^0](t,\omega) := \frac{1}{\sqrt{2\pi} \parallel h \parallel_2} \int_{\mathbb{R}} u_i^0(\tau) h * (\tau - t) e^{-\mathbf{i}\omega\tau} d\tau \tag{1}$$

The norm $\parallel h \parallel_2$ of the window function h, defined as:

$$\parallel h \parallel_2 = \sqrt{\int_{\mathbb{R}} h^2(t) dt} \tag{2}$$

is assumed to be non-zero and here we use the common convention to use the complex h^* instead of h. Similarly, we can define the time-frequency representation of the synthetics $u_i(t)$ as $\widetilde{u}_i(t,\omega) = F_h[u_i](t,\omega)$. They can also be written in exponential form:

$$\widetilde{u}_i^0(t,\omega) = \mid \widetilde{u}_i^0(t,\omega) \mid e^{\mathbf{i}\phi_i^0(t,\omega)}, \qquad \widetilde{u}_i(t,\omega) = \mid \widetilde{u}_i(t,\omega) \mid e^{\mathbf{i}\phi_i(t,\omega)} \tag{3}$$

Then we can define the envelop misfit, χ_e, and the phase misfit , χ_p, as the weighted L_2 norms of the envelop difference and phase difference, respectively:

$$\chi_e^2(u_i^0, u_i) := \int_{\mathbb{R}^2} W_e^2(t,\omega) [\mid \widetilde{u}_i(t,\omega) \mid - \mid \widetilde{u}_i^0(t,\omega) \mid]^2 dt d\omega \tag{4}$$

$$\chi_p^2(u_i^0, u_i) := \int_{\mathbb{R}^2} W_p^2(t,\omega) [\phi_i(t,\omega) - \phi_i^0(t,\omega)]^2 dt d\omega \tag{5}$$

where W_e and W_p are positive weighting functions. A equal weighted sum of this two misfits is finally used to label the data. According to the summed misfit, all available raw data is categorized into 4 classes. Class I accounts for 10% of the whole, whose misfit is smaller than 1.1 and regarded as good enough. Class II accounts for 15%, whose misfit is between 1.1 and 1.2, and needs further manual inspection. Class III accounts for 15%, whose misfit is between 1.2 and 1.3, and would be added only in latter iterations. Class IV accounts for 60%, whose misfit is larger than 1.3, and usually regarded as bad and discarded directly. These decision boundaries for different classes are empirical resulted from trial and error. Following the above algorithm, we can build up our training and testing set. Specifically, Class I to III are chosen as input. Data in Class I is assumed to be reliable and no examination is needed. Class II ($\sim$ 600 pairs of them) is determined under further visual inspection. Class III needs examination as well but not implemented in this project due to our limited time. Figure 2 shows the an example of our data pairs and their labels. Finally, the input of the network is the pairs of fixed-length time series with class labels. In the end of training, the number of seismogram pair that put into the network is 9,497 (recall that we incrementally add more data).

4 Deep Residual Network (ResNet)

While stacking more layers to increase network depth is of crucial importance to learn more features from data, it not always work. Suffering from the vanishing or exploding gradients, the deeper network would saturate in accuracy or even degrade as converging, which means that adding more layers would lead to a higher testing error as well as a higher training error. Unlike previous solution

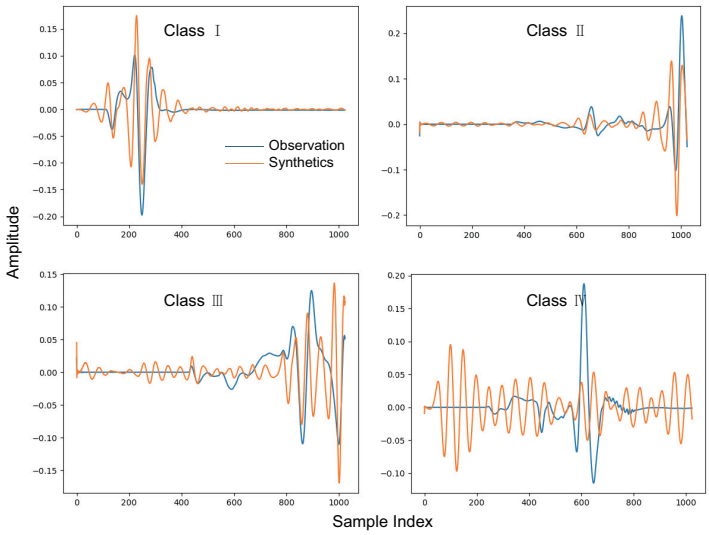

Figure 2: An example of our data pairs and their labels, the Class IV is not included in the input

using normalized initiation, Kaiming He et al. (2016) [25] proposed new architecture called deep residual network (ResNet) to deal with this problem. It contains two core idea. The first one is letting the stacked nolinear layers fit an residual mapping instead of the original underlying mapping. The second one is introducing a so-called "identity mapping shortcut connection" that skips one or more layers [42]. It has been proved to have an outstanding performance according to various benchmarks ([31], [25], [42]).

Residual Learning. Suppose we are trying to use a few stacked layers to fit a underlying mapping $H(x)$, where x denotes the inputs to the first layer. They hypothesize that those layers can also approximately learn the residual functions, and it is easier to optimize the residual mapping than the original one. So, they explicitly let the layers to learn the residual mapping $F(x) := H(x) - x$, in which case the desired function $H(x)$ would be expressed as $F(x) + x$. The degradation problem enlightens us that the stacked layers have difficulty in learning $H(x)$ in the extreme case that the added layers are identity mapping, i.e. a copy from the learned shallower layer. Alternatively, learning a residual which is zero is much simpler to approach this identity mapping.

Identity Mapping. They define the building block as:

$$\begin{aligned} y &= F(x, W_i) + x \\ &= W_2\sigma(W_1 x) + x \end{aligned} \tag{6}$$

Where x and y are the input and output vectors of the block. F denotes the residual function to learn and subscript i denotes the number of weight layer. σ represents the ReLU activation function. The operation $F + x$ is performed by a this identity mapping shortcut connection and element-wise addition. Even though this shortcut does not introduce more parameters or computation efforts, it is enough to address the degradation problem.

Pre-activation. In a following paper [26], they further point out that a pre-activation containing BN (*Batch-Normalization*) and ReLU before the weight layer is more efficient and can achieve higher training and testing accuracy compared to the traditional post-activation after weight later.

Based on these aforementioned features, we design the architecture of the residual convolutional network (figure 3) similar to that depicted in [31]. In our project, we are writing the ResNet implementation on our own, but refer to the basic structure design of CNN [3] from the Keras deep learning library, which is a high-leavel neural networks API [43]. For the time series application, we also

[3] https://keras.io/examples/cifar10_resnet/

refer to the companion repository of Fawaz et al. (2019) [4] [31]. Following the idea illustrated in Xiao et al. (2014) [44], instead of training on a static snapshot of data, we continuously add more data and more epochs to make the network grow incrementally, hoping to simulate a more practical scenario in seismology. In our architecture, we use one-hot encoding to further represent data features. The complete network includes 11 layers in total, 9 of which are convolutional layers, 1 of which is a global average pooling layer and 1 left is a softmax classifier layer to give final classes (4 as mentioned). The 9 convolutional layers are equally distributed in 3 blocks. In each block, number of filters (also the number of extracted features) are fixed to 64 in block 1, 128 in block 2 and 128 in block 3. The mini-batch size of data that fed into the network is 16. Each convolutional layer is preceded by the batch normanization and ReLU activation.

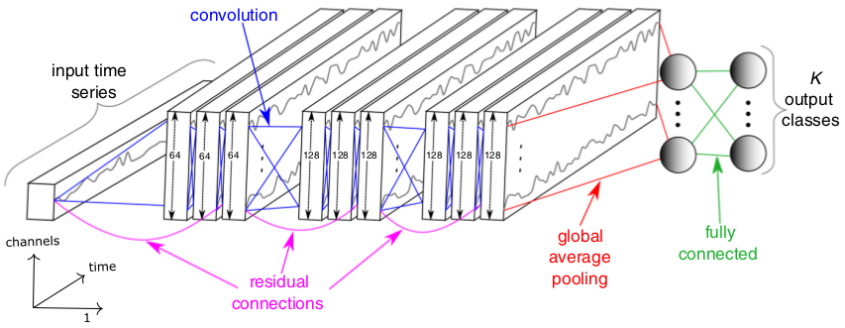

Figure 3: A typical ResNet architecture for seismogram time series classification from [31] which is based on [34]

5 Evaluation Metrics

We plan to use 3 steps to evaluate our result.

1. Compare the results of our product with the aforementioned selection (labeling) algorithm.
2. Identify contradictory results.
   (a) Perform Fourier analysis to exclude results that are obviously wrong.
   (b) Manually verify result.
3. Add the seismogram to the inversion and see whether strange patterns appear on our result (which is one of the ways we find out false decisions in our current algorithm).

Due to time limits, we only performed manual verification of part of the contradictory results.

6 Results

The output of this network is the class label predictions of the data pairs (observation and synthetics). Figure 4 shows the accuracy and loss of our default configuration (4000 testing set, 16 batches, 0.001 learning rate). The overall behavior of accuracy and loss looks similar to the result of some other studies [34]. For a cleaner representation of the result, we've decided to visualize the upper turning points of the upper turning points of the accuracy afterwards.

We have tried to use different batch sizes to train the model, and the resulting accuracy and loss is shown in the left panel of figure 4. Similarly, we try to tweak the learning rate to see the accuracy and loss change, and show them as the right panel of figure 4. According this figure, we can see that with the increase of batch size (reducing batch number), the accuracy of the result increases, but the time consumption increases as well. A batch number of 64 seems to be a good choice. Learning rate does not affect the computational time and the best learning rate is 10^-3.

[4] https://github.com/hfawaz/dl-4-tsc

We select several combinations of parameters in the network architecture to see how the performance would vary with these parameters and the result is shown in figure 5. From this figure, we can see that all parameters would have a negligible effect on the performance when the epoch is increasing to certain value (~ 200), i.e. the accuracy and loss is converging when reaching to this epoch for all parameter combinations.

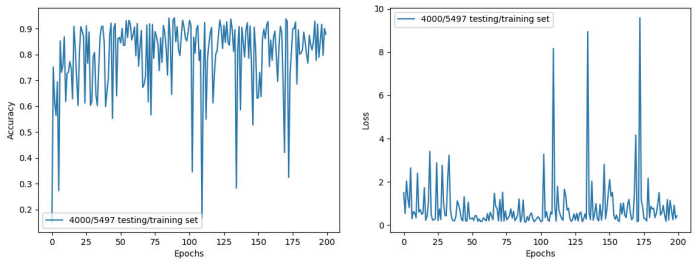

Figure 4: The variation of accuracy and loss with the epoch using the initial parameters in the architecture

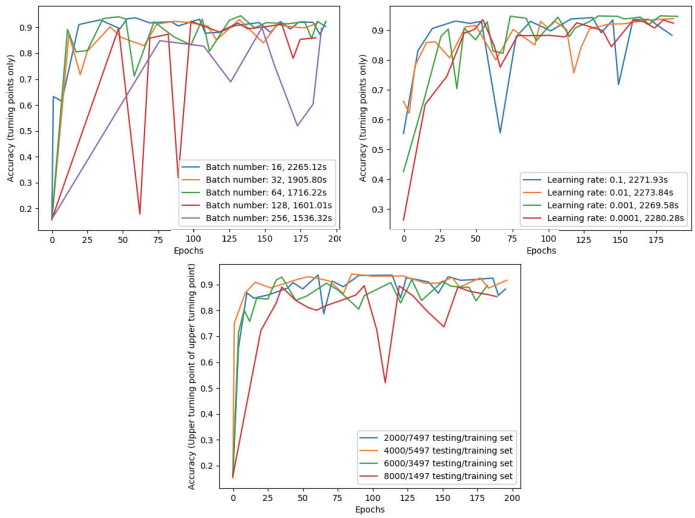

Figure 5: The resulting accuracy by changing the parameters in the architecture. Up left: batch size; Up right: learning rate; Bottom: test/train division

As a further examination for this new algorithm, we manually inspect some results, especially those suspicious ones. Inevitably, this network has its own disadvantage, such as some obviously wrong predictions, but we also find some examples that show its superiority over the original time-frequency misfit algorithm (see figure 6). In the left panel, the pair is predicted as Class III by ResNet which means not very suitable for inversion, but is determined as Class II by time-frequency misfit, and in the end is examined as Class III by human eyes. To the contrary, in the right panel, the pair is predicted as Class II by ResNet which means acceptable, but is determined as Class III by time-frequency misfit, and is examined as Class II by human eyes. Therefore, it is possible that this method is already beyond the previous algorithm, given that we are giving the decision boundaries empirically in time-frequency misfit.

7 Discussion

Due to specialty of physics behind our data, there is still plenty of space to improve this method. The first potential problem is the seismograms used in training and testing set. ResNet needs a lot of data for training, the more the better. The volume we already feed with may not be enough

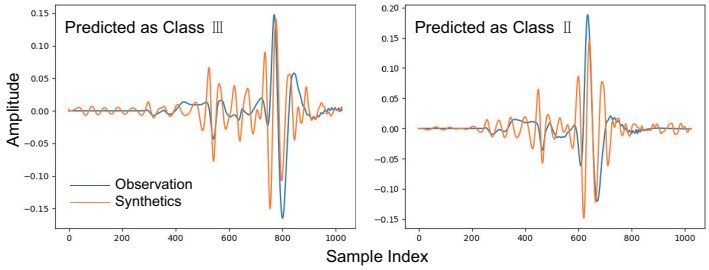

Figure 6: Two examples that show the superiority of this network. Both class labels predicted by the network are different from time-frequency misfit, but examined to be true by human eyes

yet. But increasing the data volume will not necessarily improve the results unless the diversity is also increasing, which means different kinds of good data as well as bad data are required in order to contain more features for learning. Diversity would theoretically reduce false positives and thus increase the accuracy. During the training, we try to change the ratio of test and train set, and we could see the difference like we said in the results. In the current data-preprocessing, we are using the time-frequency misfit algorithm to generate label of classes for seismograms based on the assumption that they are reliable. However, it may not hold true any longer if the algorithm misses some hidden features of seismic data, which coincidentally recognized by the ResNet. But we demand a new way to test this possibility because the current algorithm is also one of the evaluation metrics. Moreover, we are aware that a time series like seismogram has different representations. For example, instead of using Fourier transform, other transform like Wavelet transform is also doable and worth to try in the future.

In the current architecture we use in this project, the number of convolutional layers is 9 and number of features is 64 (128;128) in each layer. Since we keep adding more data and more epochs, it is obvious that the min-batch size of input data would affect both the training speed and performance. And it seems from the result that using the batch number of 64 would be a better balance between speed and accuracy. Learning rate is also influencing our result. If the learning rate is too high, the model will be changed too aggressively which harms accuracy. If the learning rate is too low, the model will also change too slowly. Besides, the number of features extracted in each convolutional block is important as well. Introducing more features would increase computation cost but not guarantee a better result. Lastly, it might be promising to add more layers to the network, because training a 1,000-layer deep ResNet is shown to exceed its shallower counterpart now [26].

As we reviewed before, there are many other machine learning methods out there. Fawaz et al. (2019) [31] showed that fully convolutional neural network also worked well. Bergen et al. (2019) [38] summarized that methods like support vector machine, random forest already have satisfying applications.

8 Conclusion

According to our implementation, using automated Resnet classification is feasible for data selection during the data processing stage before FWI. Resnet has achieved an acceptable accuracy in finding qualified data for FWI, and its performance is already better than our previous algorithm. Several parameters in the architecture design, such as mini-batch size, number of features and network depth, can be optimized to get a better result and so does the input itself, i.e. data-preprocessing. Methods other than ResNet still deserve a try since ResNet may not be the best model in general seismology. Although this project is rather preliminary, it might illuminate a broader application of ResNet in the future of Seismology, given the big data era is coming. We would regard this as our alpha version and keep developing it in the future.

References

[1] Albert Tarantola. Inversion of seismic reflection data in the acoustic approximation. *Geophysics*, 49(8):1259–1266, 1984.

[2] Jean Virieux and Stéphane Operto. An overview of full-waveform inversion in exploration geophysics. *Geophysics*, 74(6):WCC1–WCC26, 2009.

[3] Jeffrey Park, Kent Anderson, Richard Aster, Rhett Butler, Thorne Lay, and David Simpson. Global seismographic network records the great sumatra-andaman earthquake. *Eos, Transactions American Geophysical Union*, 86(6):57–61, 2005.

[4] Peter Mora. Nonlinear two-dimensional elastic inversion of multioffset seismic data. *Geophysics*, 52(9):1211–1228, 1987.

[5] R Gerhard Pratt. Seismic waveform inversion in the frequency domain, part 1: Theory and verification in a physical scale model. *Geophysics*, 64(3):888–901, 1999.

[6] C Ravaut, S Operto, L Improta, J Virieux, A Herrero, and P Dell'Aversana. Multiscale imaging of complex structures from multifold wide-aperture seismic data by frequency-domain full-waveform tomography: Application to a thrust belt. *Geophysical Journal International*, 159(3):1032–1056, 2004.

[7] Jeroen Tromp, Carl Tape, and Qinya Liu. Seismic tomography, adjoint methods, time reversal and banana-doughnut kernels. *Geophysical Journal International*, 160(1):195–216, 2005.

[8] Jerome R Krebs, John E Anderson, David Hinkley, Ramesh Neelamani, Sunwoong Lee, Anatoly Baumstein, and Martin-Daniel Lacasse. Fast full-wavefield seismic inversion using encoded sources. *Geophysics*, 74(6):WCC177–WCC188, 2009.

[9] Andreas Fichtner. *Full seismic waveform modelling and inversion*. Springer Science & Business Media, 2010.

[10] Hafedh Ben-Hadj-Ali, Stéphane Operto, and Jean Virieux. An efficient frequency-domain full waveform inversion method using simultaneous encoded sources. *Geophysics*, 76(4):R109–R124, 2011.

[11] Dimitri Komatitsch and Jeroen Tromp. Introduction to the spectral element method for three-dimensional seismic wave propagation. *Geophysical journal international*, 139(3):806–822, 1999.

[12] Dimitri Komatitsch and Jeroen Tromp. Spectral-element simulations of global seismic wave propagationi. validation. *Geophysical Journal International*, 149(2):390–412, 2002.

[13] Dimitri Komatitsch and Jeroen Tromp. Spectral-element simulations of global seismic wave propagationii. three-dimensional models, oceans, rotation and self-gravitation. *Geophysical Journal International*, 150(1):303–318, 2002.

[14] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

[15] Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. Phoneme recognition using time-delay neural networks. *Backpropagation: Theory, Architectures and Applications*, pages 35–61, 1995.

[16] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[17] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[18] Kyoung-Su Oh and Keechul Jung. Gpu implementation of neural networks. *Pattern Recognition*, 37(6):1311–1314, 2004.

[19] Dan Claudiu Ciresan, Ueli Meier, Jonathan Masci, Luca Maria Gambardella, and Jürgen Schmidhuber. Flexible, high performance convolutional neural networks for image classification. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[20] Dan Cireşan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. *arXiv preprint arXiv:1202.2745*, 2012.

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[22] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[23] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[24] Cs231n convolutional neural networks for visual recognition. `http://cs231n.github.io/convolutional-networks/`. Accessed: 2019-04-22.

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[27] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[28] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer, 2016.

[29] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[30] Andreas Veit, Michael J Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. In *Advances in neural information processing systems*, pages 550–558, 2016.

[31] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, pages 1–47, 2019.

[32] Zhicheng Cui, Wenlin Chen, and Yixin Chen. Multi-scale convolutional neural networks for time series classification. *arXiv preprint arXiv:1603.06995*, 2016.

[33] John Cristian Borges Gamboa. Deep learning for time-series analysis. *arXiv preprint arXiv:1701.01887*, 2017.

[34] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*, pages 1578–1585. IEEE, 2017.

[35] Benjamin K Holtzman, Arthur Paté, John Paisley, Felix Waldhauser, and Douglas Repetto. Machine learning reveals cyclic changes in seismic source spectra in geysers geothermal field. *Science advances*, 4(5):eaao2929, 2018.

[36] Zachary E. Ross, Men-Andrin Meier, and Egill Hauksson. P wave arrival picking and first-motion polarity determination with deep learning. *Journal of Geophysical Research: Solid Earth*, 123(6):5120–5129, 2018.

[37] Daigo Shoji, Rina Noguchi, Shizuka Otsuki, and Hideitsu Hino. Classification of volcanic ash particles using a convolutional neural network and probability. *Scientific reports*, 8(1):8111, 2018.

[38] Karianne J. Bergen, Paul A. Johnson, Maarten V. de Hoop, and Gregory C. Beroza. Machine learning for data-driven discovery in solid earth geoscience. *Science*, 363(6433), 2019.

[39] Kasra Hosseini and Karin Sigloch. obspydmt: A python toolbox for retrieving and processing of large seismological datasets. *Solid Earth*, 8, 2017.

[40] Miriam Kristeková, Jozef Kristek, Peter Moczo, and Steven M Day. Misfit criteria for quantitative comparison of seismograms. *Bulletin of the seismological Society of America*, 96(5):1836–1850, 2006.

[41] Miriam Kristeková, Jozef Kristek, and Peter Moczo. Time-frequency misfit and goodness-of-fit criteria for quantitative comparison of time signals. *Geophysical Journal International*, 178(2):813–825, 2009.

[42] Vincent Fung. An overview of resnet and its variants. `https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035`. Accessed: 2019-04-22.

[43] François Chollet et al. Keras. `https://keras.io`, 2015.

[44] Tianjun Xiao, Jiaxing Zhang, Kuiyuan Yang, Yuxin Peng, and Zheng Zhang. Error-driven incremental learning in deep convolutional neural network for large-scale image classification. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 177–186. ACM, 2014.