UNDERSTANDING AND IMPROVING TRANSFORMER FROM A MULTI-PARTICLE DYNAMIC SYSTEM POINT OF VIEW

Anonymous authors
Paper under double-blind review

ABSTRACT

The Transformer architecture is widely used in natural language processing. Despite its success, the design principle of the Transformer remains elusive. In this paper, we provide a novel perspective towards understanding the architecture: we show that the Transformer can be mathematically interpreted as a *numerical Ordinary Differential Equation (ODE) solver for a convection-diffusion equation in a multi-particle dynamic system*. In particular, how words in a sentence are abstracted into contexts by passing through the layers of the Transformer can be interpreted as approximating multiple particles' movement in the space using the Lie-Trotter splitting scheme and the Euler's method. Inspired from such a relationship, we propose to replace the Lie-Trotter splitting scheme by the more accurate Strang-Marchuk splitting scheme and design a new network architecture called Macaron Net. Through extensive experiments, we show that the Macaron Net is superior to the Transformer on both supervised and unsupervised learning tasks.

1 INTRODUCTION

The Transformer is one of the most commonly used neural network architectures in natural language processing. Variants of the Transformer have achieved state-of-the-art performance in many tasks including language modeling Dai et al. (2019); Al-Rfou et al. (2018) and machine translation Vaswani et al. (2017); Dehghani et al. (2018); Edunov et al. (2018). Transformer-based unsupervised pre-trained models also show impressive performance in many downstream tasks Radford et al. (2019); Devlin et al. (2018); Peters et al. (2018). Although the Transformer has demonstrated promising results in many tasks, its design principle is not fully understood in previous works, and thus the strength of the architecture is not fully exploited.

In this paper, we provide a novel perspective towards understanding the architecture. Inspired by the relationship between the ODE and neural networks Weinan (2017); Lu et al. (2017); Haber & Ruthotto (2017); Chen et al. (2018a); Zhang et al. (2019); Sonoda & Murata (2019); Thorpe & van Gennip (2018), we show that the Transformer layers can be naturally interpreted as a numerical ODE solver for a first-order convection-diffusion equation in a multi-particle dynamic system (MPDS). To be more specific, the self-attention sub-layer, which transforms the semantics at one position by attending over all other positions, corresponds to the diffusion term; The position-wise FFN sub-layer, which is applied to each position separately and identically, corresponds to the convection term. The number of stacked layers in the Transformer corresponds to the time dimension in ODE. In this way, the stack of self-attention sub-layers and position-wise FFN sub-layers with residual connections can be viewed as solving the ODE problem numerically using the Lie-Trotter splitting scheme Geiser (2009) and the Euler's method Ascher & Petzold (1998). By this interpretation, we have a novel understanding of learning contextual representations of a sentence using the Transformer: the feature (a.k.a, embedding) of words in a sequence can be considered as initial positions of a collection of particles, and the latent representations abstracted in stacked Transformer layers can be viewed as the location of particles moving in a high-dimensional space at different time steps.

Such an interpretation not only provides a new perspective on the Transformer but also inspires us to design new structures by leveraging the rich literature of numerical analysis. The Lie-Trotter splitting scheme is simple but not accurate and often leads to high approximation error Geiser (2009). The Strang-Marchuk splitting scheme Strang (1968) is developed to reduce the approximation error by a simple modification to the Lie-Trotter splitting scheme and is theoretically more accurate. This observation motivates us to design a new neural network architecture *Macaron Net* based on the Strang-Marchuk splitting scheme. Our extensive experiments on both supervised and unsupervised learning tasks shows that the Macaron Net can achieve higher accuracy than the Transformer on all tasks which, in a way, is consistent with the ODE theory.

2 MAIN RESULTS

2.1 BACKGROUND ON THE TRANSFORMER ARCHITECTURE

We first briefly introduce the Transformer architecture and more detailed information can be found in previous works Vaswani et al. (2017); Devlin et al. (2018). A Transformer layer operates on a sequence of vectors and outputs a new sequence of the same shape. Each layer can be decomposed into two sub-layers: a (multi-head) self-attention sub-layer and a position-wise feed-forward network sub-layer. Residual connection He et al. (2016) and layer normalization Lei Ba et al. (2016) are employed for both sub-layers.

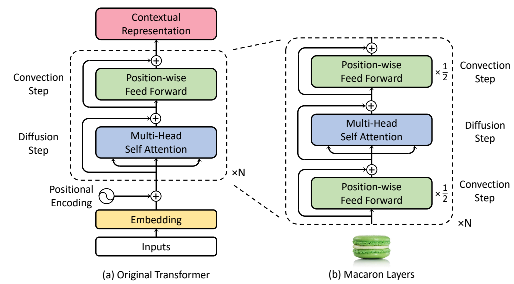

Figure 1: The Transformer and our Macaron architectures.

Self-attention sub-layer The attention mechanism can be formulated as querying a dictionary with key-value pairs (Vaswani et al., 2017), e.g., Attention$(Q, K, V) = \text{softmax}(QK^T/\sqrt{d_{model}}) \cdot V$, where d_{model} is the dimensionality of the hidden representations and Q (Query), K (Key), V (Value) are specified as the hidden representations of the previous layer in the so-called *self-attention* sub-layers in the Transformer architecture. The multi-head variant of attention is more popularly used which allows the model to jointly attend to information from different representation subspaces.

Position-wise FFN sub-layer In addition to the self-attention sub-layer, each Transformer layer also contains a fully connected feed-forward network, which is applied to each position separately and identically. This feed-forward network consists of two linear transformations with an activation function σ in between. Specially, given vectors $h_1, \ldots, h_n$, a position-wise FFN sub-layer transforms each h_i as FFN$(h_i) = \sigma(h_i W_1 + b_1)W_2 + b_2$, where W_1, W_2, b_1 and b_2 are parameters.

2.2 PHYSICAL INTERPRETATION OF THE TRANSFORMER AS A MULTI-PARTICLE ODE SOLVER

Understanding the dynamics of multiple particles' movements in space is one of the important problems in physics, especially in fluid mechanics and astrophysics Moulton (2012). The behavior of each particle is usually modeled by two factors: The first factor concerns about the mechanism of its movement regardless of other particles, e.g., caused by an external force outside of the system, which is usually referred to as the convection; The second factor concerns about the movement resulting from other particles, which is usually referred to as the diffusion. Mathematically, assume there are n particles in d-dimensional space. Denote $x_i(t) \in \mathbb{R}^d$ as the location of i-th particle at time t. The dynamics of particle i can be formulated as

$$\frac{\mathrm{d}x_i(t)}{\mathrm{d}t} = F(x_i(t), [x_1(t), \cdots, x_n(t)], t) + G(x_i(t), t),$$
$$x_i(t_0) = w_i, \quad i = 1, \ldots, n. \tag{1}$$

Function $F(x_i(t), [x_1(t), \cdots, x_n(t)], t)$ represents the diffusion term which characterizes the interaction between the particles. $G(x, t)$ is a function which takes a location x and time t as input and represents the convection term. As we can see, there are two coupled terms in the right-hand side of Eqn (1) describing different physical phenomena. The splitting method is a prevailing way of solving such coupled differential equations that can be decomposed into a sum of differential operators McLachlan & Quispel (2002). The Lie-Trotter splitting scheme Geiser (2009) is the simplest splitting method. It splits the right-hand side of Eqn (1) into function $F(\cdot)$ and $G(\cdot)$ and solves the individual dynamics alternatively. More precisely, to compute $x_i(t + \gamma)$ from $x_i(t)$, the Lie-Trotter

Table 1: Translation performance (BLEU) on IWSLT14 De-En and WMT14 En-De testsets.

Method	IWSLT14 De-En small	WMT14 En-De base	WMT14 En-De big
Transformer Vaswani et al. (2017)	34.4	27.3	28.4
Universal Transformer Dehghani et al. (2018)	/	**28.9**	/
Scaling NMT Ott et al. (2018)	/	/	29.3
Dynamic Conv Wu et al. (2019)	35.2	/	29.7
Macaron Net	**35.4**	**28.9**	**30.2**

splitting scheme with the Euler's method reads as

$$\tilde{x}_i(t) = x_i(t) + \gamma F(x_i(t), [x_1(t), x_2(t), \cdots, x_n(t)], t), \tag{2}$$

$$x_i(t+\gamma) = \tilde{x}_i(t) + \gamma G(\tilde{x}_i(t), t). \tag{3}$$

We reformulate the two sub-layers of the Transformer in order to match its form with the ODE described above (more discussions on the close relationship between ODE and neural network can be found in the Appendix). Denote $x_l = (x_{l,1}, \ldots, x_{l,n})$ as the input to the l-th Transformer layer, where n is the sequence length and $x_{l,i}$ is a real-valued vector in $\mathbb{R}^d$ for any i. Denote $\tilde{x}_{l,i}$ as the output of the (multi-head) self-attention sub-layer at position i with residual connections. The computation of $\tilde{x}_{l,i}$ can be written as $\tilde{x}_{l,i} = x_{l,i} + \text{Concat}(\text{head}_1, ..., \text{head}_H)W^{O,l}$, where $\text{head}_k = \sum_{j=1}^{n} \alpha_{ij}^{(k)} [x_{l,j} W_k^{V,l}] = \sum_{j=1}^{n} (\exp(e_{ij}^{(k)}) / (\sum_{q=1}^{n} \exp(e_{iq}^{(k)}))) x_{l,j} W_k^{V,l}$, and $e_{ij}^{(k)}$ is computed as the dot product of input $x_{l,i}$ and $x_{l,j}$ with linear projection matrices $W_k^{Q,l}$ and $W_k^{K,l}$. Therefore, we can generally reformulate the multi-head attention as $\tilde{x}_{l,i} = x_{l,i} + \text{MultiHeadAtt}_{W_{att}^l}(x_{l,i}, [x_{l,1}, x_{l,2}, \cdots, x_{l,n}])$, where W_{att}^l denotes all trainable parameters in the l-th self-attention sub-layer. Similarly, we can formulate the position-wise feed-forward sub-layer as $x_{l+1,i} = \tilde{x}_{l,i} + \text{FFN}_{W_{ffn}^l}(\tilde{x}_{l,i})$, where W_{ffn}^l denotes all trainable parameters in the l-th position-wise FFN sub-layer. Combining above equations, we reformulate the Transformer layers as

$$\tilde{x}_{l,i} = x_{l,i} + \text{MultiHeadAtt}_{W_{att}^l}(x_{l,i}, [x_{l,1}, x_{l,2}, \cdots, x_{l,n}]), \tag{4}$$

$$x_{l+1,i} = \tilde{x}_{l,i} + \text{FFN}_{W_{ffn}^l}(\tilde{x}_{l,i}). \tag{5}$$

We can see that the Transformer layers (Eqn (4-5)) resemble the multi-particle ODE solver in Section 2.2 (Eqn (2-3)), which grants a physical interpretation of natural language processing and provides a new perspective on the Transformer architecture. The self-attention sub-layer is viewed as a diffusion term which characterizes the particle interactions while the position-wise feed-forward networks is viewed as a convection term. The two terms together naturally form the convection-diffusion equation in physics.

2.3 Improving Transformer Via Strang-Marchuk Splitting Scheme

The Lie-Trotter splitting scheme solves the dynamics of $F(\cdot)$ and $G(\cdot)$ alternatively and exclusively in that order. This inevitably brings bias and leads to higher local truncation errors Geiser (2009). To mitigate the bias, the Strang-Marchuk splitting scheme Strang (1968) is more widely used. Mathematically, to compute $x_i(t+\gamma)$ from $x_i(t)$, the Strang-Marchuk splitting scheme reads as

$$\tilde{x}_i(t) = x_i(t) + \frac{\gamma}{2} G(x_i(t), t), \tag{6}$$

$$\hat{x}_i(t) = \tilde{x}_i(t) + \gamma F(\tilde{x}_i(t), [\tilde{x}_1(t), \tilde{x}_2(t), \cdots, \tilde{x}_n(t)], t), \tag{7}$$

$$x_i(t+\gamma) = \hat{x}_i(t) + \frac{\gamma}{2} G\left(\hat{x}_i(t), t + \frac{\gamma}{2}\right). \tag{8}$$

We can see from Eqn (6-8) that the Strang-Marchuk splitting scheme uses a three-step process to solve the ODE. Mapped to neural network design, by replacing function γF and γG by MultiHeadAtt and FFN, the Strang-Marchuk splitting scheme (together with the Euler's method) suggests there should also be three sub-layers instead of the two sub-layers in Transformer. Each hidden vector at different positions will first pass through the first position-wise FFN sub-layer with a half-step residual connection ("$\frac{1}{2}$" in Eqn (6 & 8)), and then the output vectors will be feed into a self-attention sub-layer. At last, the vectors outputted from the self-attention sub-layer will be put into another half-step FFN sub-layer. We call the resulted "Macaron"-like structure *Macaron Net*, as in Figure 1(b).

Table 2: Test results on the GLUE benchmark (except WNLI).

Method	CoLA	SST-2	MRPC	STS-B	QQP	MNLI-m/mm	QNLI	RTE	**GLUE**
OpenAI GPT Radford et al.	47.2	93.1	87.7/83.7	85.3/84.8	70.1/88.1	80.7/80.6	87.2	69.1	76.9
BERT `base` Devlin et al. (2018)	52.1	93.5	**88.9/84.8**	87.1/85.8	**71.2/89.2**	84.6/83.4	90.5	66.4	78.3
BERT `base` (ours)	52.8	92.8	87.3/83.0	81.2/80.0	70.2/88.4	84.4/83.7	90.4	64.9	77.4
Macaron Net `base`	**57.6**	**94.0**	88.4/84.4	**87.5/86.3**	70.8/89.0	**85.4/84.5**	**91.6**	**70.5**	**79.7**

3 EXPERIMENTS

We test our proposed Macaron architectures in both supervised and unsupervised learning setting. For supervised learning setting, we use IWLST14 De-En and WMT14 En-De machine translation datasets. For unsupervised learning setting, we pretrain the model using the same method as in BERT Devlin et al. (2018). More information can be found in Appendix.

3.1 EXPERIMENT SETTINGS

Machine Translation For the WMT14 dataset, we use Transformer with the `base` and the `big` settings Vaswani et al. (2017). Both of them consist of a 6-layer encoder and 6-layer decoder. The size of the hidden nodes and embeddings are set to 512 for `base` and 1024 for `big`. For the IWSLT14 dataset, we use the `small` setting, whose size of hidden states and embeddings is set to 512. For each setting (`base`, `big` and `small`), we replace all Transformer layers by the Macaron layers. To make a fair comparison, we reduce the dimensionality of the inner-layer of the two FFN sub-layers in the Macaron layers by half. By doing this, the `base`, `big` and `small` Macaron have the same number of parameters as the `base`, `big` and `small` Transformer respectively.

Unsupervised Pretraining BERT Devlin et al. (2018) is the current state-of-the-art pre-trained contextual representation model based on a multi-layer Transformer encoder architecture and trained by masked language modeling and next-sentence prediction tasks. We compare our proposed Macaron Net with the `base` setting from the original BERT paper Devlin et al. (2018), which consists of 12 Transformer layers. The size of hidden states and embeddings are set to 768, and the number of attention heads is set to 12. Similarly, we replace the Transformer layers in BERT `base` by the Macaron layers and reduce the dimensionality of the inner-layer of the two FFN sub-layers by half, and thus we keep the number of parameters of our Macaron `base` same as BERT `base`.

3.2 EXPERIMENT RESULTS

Machine Translation The results for machine translation are shown in Table 1. For the IWSLT14 dataset, our Macaron `small` outperforms the Transformer `small` by 1.0 point in terms of BLEU. For the WMT14 dataset, our Macaron `base` outperforms its Transformer counterpart by 1.6 BLEU points. Our Macaron `big` outperforms the Transformer `big` by 1.8 BLEU points. Comparing with other concurrent works, the improvements in our proposed method are significant.

Unsupervised Pretraining Following Devlin et al. (2018), we evaluate all models by fine-tuning them on 8 downstream tasks in the General Language Understanding Evaluation (GLUE) benchmark Wang et al. (2019). The results are presented in Table 2. We present the results of two BERT `base` models. One is from Devlin et al. (2018), and the other is our reproduced one. We can see from the table, our proposed Macaron Net `base` outperforms all baselines in terms of the general GLUE score.

As a summary, the improvement in both machine translation and GLUE tasks well aligns with the ODE theory and our proposed architecture performs better than the Transformer in real practice.

4 CONCLUSION AND FUTURE WORK

In this paper, we interpret the Transformer as a numerical ODE solver for a convection-diffusion equation in a multi-particle dynamic system and develop new architecuture from the rich literature of ODE theory. In the future, we will explore deeper connections between the ODE theory and the Transformer models.

REFERENCES

Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-level language modeling with deeper self-attention. *arXiv preprint arXiv:1808.04444*, 2018.

Uri M Ascher and Linda R Petzold. *Computer methods for ordinary differential equations and differential-algebraic equations*, volume 61. Siam, 1998.

Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo. Magnini. The fifth PASCAL recognizing textual entailment challenge. 2009.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.

Bo Chang, Minmin Chen, Eldad Haber, and Ed H Chi. Antisymmetricrnn: A dynamical system view on recurrent neural networks. *arXiv preprint arXiv:1902.09689*, 2019.

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, pp. 6572–6583, 2018a.

Z. Chen, H. Zhang, X. Zhang, and L. Zhao. Quora question pairs. 2018b. URL `https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs`.

Zihang Dai, Zhilin Yang, Yiming Yang, William W Cohen, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. *arXiv preprint arXiv:1807.03819*, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

William B Dolan and Chris. Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the International Workshop on Paraphrasing.*, 2005.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*, 2018.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional Sequence to Sequence Learning. In *Proc. of ICML*, 2017.

Juergen Geiser. *Decomposition methods for differential equations: theory and applications*. CRC Press, 2009.

Eldad Haber and Lars Ruthotto. Stable architectures for deep neural networks. *Inverse Problems*, 34(1):014004, 2017.

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

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *ACL*, 2007.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Hector J Levesque, Ernest Davis, and Leora. Morgenstern. The Winograd schema challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning.*, volume 46, pp. 47, 2011.

Qianli Liao and Tomaso Poggio. Bridging the gaps between residual learning, recurrent neural networks and visual cortex. *arXiv preprint arXiv:1604.03640*, 2016.

Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. *arXiv preprint arXiv:1710.10121*, 2017.

Brian W Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451, 1975.

Robert I McLachlan and G Reinout W Quispel. Splitting methods. *Acta Numerica*, 11:341–434, 2002.

Forest Ray Moulton. *An introduction to celestial mechanics*. Courier Corporation, 2012.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. Scaling neural machine translation. *arXiv preprint arXiv:1806.00187*, 2018.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 311–318. Association for Computational Linguistics, 2002.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy. Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of EMNLP.*, pp. 2383–2392. Association for Computational Linguistics, 2016.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *ACL*, 2016.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP.*, pp. 1631–1642, 2013.

Sho Sonoda and Noboru Murata. Transport analysis of infinitely deep neural network. *The Journal of Machine Learning Research*, 20(1):31–82, 2019.

Gilbert Strang. On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis*, 5(3):506–517, 1968.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.

Yunzhe Tao, Qi Sun, Qiang Du, and Wei Liu. Nonlocal neural networks, nonlocal diffusion and nonlocal modeling. In *Advances in Neural Information Processing Systems*, pp. 496–506, 2018.

Matthew Thorpe and Yves van Gennip. Deep limits of residual neural networks. *arXiv preprint arXiv:1810.11741*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. 2019. In the Proceedings of ICLR.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. Neural network acceptability judgments. *arXiv preprint 1805.12471*, 2018.

E Weinan. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 5(1):1–11, 2017.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL-HLT.*, 2018.

Felix Wu, Angela Fan, Alexei Baevski, Yann N Dauphin, and Michael Auli. Pay less attention with lightweight and dynamic convolutions. *arXiv preprint arXiv:1901.10430*, 2019.

Xiaoshuai Zhang, Yiping Lu, Jiaying Liu, and Bin Dong. Dynamically unfolding recurrent restorer: A moving endpoint control method for image restoration. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=SJfZKiC5FX.

Mai Zhu, Bo Chang, and Chong Fu. Convolutional neural networks combined with runge-kutta methods. *arXiv preprint arXiv:1802.08831*, 2018.

Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *arXiv preprint arXiv:1506.06724*, 2015.

A RELATED WORKS

A.1 RELATIONSHIP BETWEEN NEURAL NETWORKS AND ODE

Recently, there are extensive studies to bridge deep neural networks with ordinary differential equations Weinan (2017); Lu et al. (2017); Haber & Ruthotto (2017); Chen et al. (2018a); Zhang et al. (2019); Sonoda & Murata (2019); Thorpe & van Gennip (2018). We here present a brief introduction to such a relationship and discuss how previous works borrow powerful tools from numerical analysis to help deep neural network design.

A first-order ODE problem is usually defined as to solve the equation (i.e., calculate $x(t)$ for any t) which satisfies the following first-order derivative and the initial condition:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = f(x,t), \quad x(t_0) = w, \tag{9}$$

in which $x(t) \in \mathbb{R}^d$ for all $t \geq t_0$. ODEs usually have physical interpretations. For example, $x(t)$ can be considered as the location of a particle moving in the d-dimensional space and the first order time derivative can be considered as the velocity of the particle.

Usually there is no analytic solution to Eqn (9) and the problem has to be solved numerically. The simplest numerical ODE solver is the Euler's method Ascher & Petzold (1998). The Euler's method discretizes the time derivative $\frac{\mathrm{d}x(t)}{\mathrm{d}t}$ by its first-order approximation $\frac{x(t_2)-x(t_1)}{t_2-t_1} \approx f(x(t_1),t_1)$. By doing so, for the fixed time horizon $T = t_0 + \gamma L$, we can estimate $x(T)$ from $x_0 \doteq x(t_0)$ by sequentially estimating $x_{l+1} \doteq x(t_{l+1})$ using

$$x_{l+1} = x_l + \gamma f(x_l, t_l) \tag{10}$$

where $l = 0, \cdots, L-1$, $t_l = t_0 + \gamma l$ is the time point corresponds to x_l, and $\gamma = (T - t_0)/L$ is the step size. As we can see, this is mathematically equivalent to the ResNet architecture Lu et al. (2017); Chen et al. (2018a): The function $\gamma f(x_l, t_l)$ can be considered as a neural-network block, and the second argument t_l in the function indicates the set of parameters in the l-th layer. The simple temporal discretization by Euler's method naturally leads to the residual connection.

Observing such a strong relationship, researchers use ODE theory to explain and improve the neural network architectures mainly designed for computer vision tasks. Lu et al. (2017); Chen et al. (2018a) show any parametric ODE solver can be viewed as a deep residual network (probably with infinite layers), and the parameters in the ODE can be optimized through backpropagation. Recent works discover that new neural networks inspired by sophisticated numerical ODE solvers can lead to better performance. For example, Zhu et al. (2018) uses a high-precision Runge-Kutta method to design a neural network, and the new architecture achieves higher accuracy. Haber & Ruthotto (2017) uses a leap-frog method to construct a reversible neural network. Liao & Poggio (2016); Chang et al. (2019) try to understand recurrent neural networks from the ODE's perspective, and Tao et al. (2018) uses non-local differential equations to model non-local neural networks.

B EXPERIMENT SETTINGS

B.1 MACHINE TRANSLATION

Dataset The training/validation/test sets of the IWSLT14 dataset contain about 153K/7K/7K sentence pairs, respectively. We use a vocabulary of 10K tokens based on a joint source and target byte pair encoding (BPE) Sennrich et al. (2016). For WMT14 dataset, we replicate the setup of Vaswani et al. (2017), which contains 4.5M training parallel sentence pairs. Newstest2014 is used as the test set, and Newstest2013 is used as the validation set. The 37K vocabulary for WMT14 is based on a joint source and target BPE factorization.

Model For the WMT14 dataset, the basic configurations of the Transformer architecture are the `base` and the `big` settings Vaswani et al. (2017). Both of them consist of a 6-layer encoder and 6-layer decoder. The size of the hidden nodes and embeddings are set to 512 for `base` and 1024 for `big`. The number of heads are 8 for `base` and 16 for `big`. Since the IWSLT14 dataset is much smaller than the WMT14 dataset, the `small` setting is usually used, whose size of hidden states and embeddings is set to 512 and the number of heads is set to 4. For all settings, the dimensionality of the inner-layer of the position-wise FFN is four times of the dimensionality of the hidden states.

For each setting (`base`, `big` and `small`), we replace all Transformer layers by the Macaron layers and obtain the `base`, `big` and `small` Macaron, each of which contains two position-wise feed-forward sub-layers in a layer. The translation model is based on the encoder-decoder framework. In the Transformer, the decoder layer has a third sub-layer which performs multi-head attention over the output of the encoder stack (encoder-decoder-attention) and a mask to prevent positions from attending to subsequent positions. In our implementation of Macaron decoder, we also use masks and split the FFN into two sub-layers and thus our decoder layer is (FFN, self-attention, encoder-decoder-attention and FFN).

To make a fair comparison, we set the dimensionality of the inner-layer of the two FFN sub-layers in the Macaron layers to two times of the dimensionality of the hidden states. By doing this, the `base`, `big` and `small` Macaron have the same number of parameters as the `base`, `big` and `small` Transformer respectively.

Optimizer and training We use the Adam optimizer and follow the optimizer setting and learning rate schedule in Vaswani et al. (2017). For the `big` setting, we enlarge the batch size and learning rate as suggested in Ott et al. (2018) to accelerate training. We employ label smoothing of value $\epsilon_{ls} = 0.1$ (Szegedy et al., 2016) in all experiments. Models for WMT14/IWSLT14 are trained on 4/1 NVIDIA P40 GPUs respectively. Our code is based on the open-sourced `fairseq` Gehring et al. (2017) code base in PyTorch toolkit.

Evaluation We use BLEU[1] Papineni et al. (2002) as the evaluation measure for machine translation. Following common practice, we use tokenized case-sensitive BLEU and case-insensitive BLEU for WMT14 En-De and IWSLT14 De-En respectively. During inference, we use beam search with beam size 4 and length penalty 0.6 for WMT14, and beam size 5 and length penalty 1.0 for IWSLT14, following Vaswani et al. (2017).

B.2 Unsupervised Pretraining

Pre-training dataset We follow Devlin et al. (2018) to use English Wikipedia corpus and BookCorpus for pre-training. As the dataset BookCorpus Zhu et al. (2015) is no longer freely distributed. We follow the suggestions from Devlin et al. (2018) to crawl and collect BookCorpus[2] on our own. The concatenation of two datasets includes roughly 3.4B words in total, which is comparable with the data corpus used in Devlin et al. (2018). We first segment documents into sentences with Spacy;[3] Then, we normalize, lower-case, and tokenize texts using MosesKoehn et al. (2007) and apply BPESennrich et al. (2016). We randomly split documents into one training set and one validation set. The training-validation ratio for pre-training is 199:1.

Model We compare our proposed Macaron Net with the `base` setting from the original BERT paper Devlin et al. (2018), which consists of 12 Transformer layers. The size of hidden states and embeddings are set to 768, and the number of attention heads is set to 12. Similarly, we replace the Transformer layers in BERT `base` by the Macaron layers and reduce the dimensionality of the inner-layer of the two FFN sub-layers by half, and thus we keep the number of parameters of our Macaron `base` as the same as BERT `base`.

Optimizer and training We follow Devlin et al. (2018) to use two tasks to pretrain our model. One task is masked language modeling, which masks some percentage of the input tokens at random, and then requires the model to predict those masked tokens. Another task is next sentence prediction, which requires the model to predict whether two sentences in a given sentence pair are consecutive. We use the Adam optimizer and and follow the optimizer setting and learning rate schedule in Devlin et al. (2018) and trained the model on 4 NVIDIA P40 GPUs.

[1] `https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl`

[2] `https://www.smashwords.com/`

[3] `https://spacy.io`

B.3 GLUE DATASET

We provide a brief description of the tasks in the GLUE benchmark Wang et al. (2019) and our fine-tuning process on the GLUE datasets.

CoLA The Corpus of Linguistic Acceptability Warstadt et al. (2018) consists of English acceptability judgments drawn from books and journal articles on linguistic theory. The task is to predict whether an example is a grammatical English sentence. The performance is evaluated by Matthews correlation coefficient Matthews (1975).

SST-2 The Stanford Sentiment Treebank Socher et al. (2013) consists of sentences from movie reviews and human annotations of their sentiment. The task is to predict the sentiment of a given sentence (positive/negative). The performance is evaluated by the test accuracy.

MRPC The Microsoft Research Paraphrase Corpus Dolan & Brockett (2005) is a corpus of sentence pairs automatically extracted from online news sources, with human annotations for whether the sentences in the pair are semantically equivalent, and the task is to predict the equivalence. The performance is evaluated by both the test accuracy and the test F1.

STS-B The Semantic Textual Similarity Benchmark Cer et al. (2017) is a collection of sentence pairs drawn from news headlines, video and image captions, and natural language inference data. Each pair is human-annotated with a similarity score from 1 to 5; the task is to predict these scores. The performance is evaluated by Pearson and Spearman correlation coefficients.

QQP The Quora Question Pairs[4] Chen et al. (2018b) dataset is a collection of question pairs from the community question-answering website Quora. The task is to determine whether a pair of questions are semantically equivalent. The performance is evaluated by both the test accuracy and the test F1.

MNLI The Multi-Genre Natural Language Inference Corpus Williams et al. (2018) is a crowd-sourced collection of sentence pairs with textual entailment annotations. Given a premise sentence and a hypothesis sentence, the task is to predict whether the premise entails the hypothesis (*entailment*), contradicts the hypothesis (*contradiction*), or neither (*neutral*). The performance is evaluated by the test accuracy on both *matched* (in-domain) and *mismatched* (cross-domain) sections of the test data.

QNLI The Question-answering NLI dataset is converted from the Stanford Question Answering Dataset (SQuAD) Rajpurkar et al. (2016) to a classification task. The performance is evaluated by the test accuracy.

RTE The Recognizing Textual Entailment (RTE) datasets come from a series of annual textual entailment challenges Bentivogli et al. (2009). The task is to predict whether sentences in a sentence pair are entailment. The performance is evaluated by the test accuracy.

WNLI The Winograd Schema Challenge Levesque et al. (2011) is a reading comprehension task in which a system must read a sentence with a pronoun and select the referent of that pronoun from a list of choices. We follow Devlin et al. (2018) to skip this task in our experiments, because few previous works do better than predicting the majority class for this task.

Fine-tuning on GLUE tasks To fine-tune the models, following Devlin et al. (2018), we search the optimization hyperparmaters in a search space including different batch sizes (16/32), learning rates (5e-3/3e-5), number of epochs (3/4/5), and a set of different random seeds. We select the model for testing according to their performance on the development set.

[4] `https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs`

Test data Note that the GLUE dataset distribution does not include the Test labels, and we only made a single GLUE evaluation server submission[5] for each of our models.

[5] https://gluebenchmark.com