Designing Transformers with Kernel Methods

Grégoire Mialon

Inria Paris

MILA - DIRO, Université de Montreal.

What I am interested in

Kernel methods and deep learning

- G. Mialon*, D. Chen*, M. Selosse*, J. Mairal. Structural Graph Transformers (to appear on arXiv).
- G. Mialon*, D. Chen*, A. d'Aspremont, J. Mairal. A Trainable Optimal Transport Embedding for Feature Aggregation and its Relationship to Attention (ICLR, 2021).
- A. Bietti*, G. Mialon*, D. Chen, J. Mairal. A Kernel Perspective for Regularizing Deep Neural Networks (ICML, 2019).

Convex optimization

- G. Mialon, A. d'Aspremont, J. Mairal. Screening Data Points in Empirical Risk Minimization via Ellipsoidal Regions and Safe Loss Functions (AISTATS, 2020).

Causal inference

What I want to talk about today

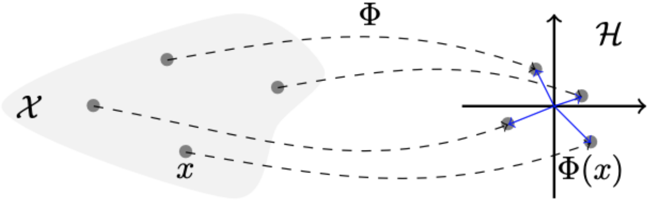
Kernel methods and transformers

- G. Mialon*, D. Chen*, M. Selosse*, J. Mairal. Structural Graph Transformers (to appear on arXiv).
- G. Mialon*, D. Chen*, A. d'Aspremont, J. Mairal. A Trainable Optimal Transport Embedding for Feature Aggregation and its Relationship to Attention (ICLR, 2021).

Why kernel methods?

- Reconciling deep learning with small data regimes.
- Understanding architectures with a kernel lens.

Kernel methods



Learning with Kernel methods

- Map data x to high-dimensional space, $\Phi(x) \in \mathcal{H}$ (RKHS).
- Φ associated to a positive definite kernel K: $K(x, x') = \langle \Phi(x), \Phi(x') \rangle_{\mathcal{H}}$ (kernel trick).
- Convex optimization for learning linear decision function in the RKHS.

Transformers, self-attention, and kernel smoothing

Transformers (encoder).

- A sequence of layers processing an input set of d_{in} features X in $\mathbb{R}^{n \times d_{\text{in}}}$, and compute another set in $\mathbb{R}^{n \times d_{\text{out}}}$.
- Self-attention mechanism:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_{out}}}\right) V \in \mathbb{R}^{n \times d_{\text{out}}}, \tag{1}$$

  with $Q^\top = W_Q X^\top$ and $K^\top = W_K X^\top$ resp. query and key matrices, $V^\top = W_V X^\top$ the value matrix, and W_Q, W_K, W_V in $\mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ learned projection matrices.
- During forward pass, feature map X updated via:

$$X = X + \text{Attention}(Q, K, V).$$

- LayerNorm and "element-wise" feed-forward.

Transformers, self-attention, and kernel smoothing

Self-attention as a kernel smoothing.

- We can rewrite self-attention:

$$\mathrm{Attention}(Q, K, V)_i = \sum_{j=1}^{n} \frac{exp\left(\frac{Q_i K_j^\top}{\sqrt{d_{out}}}\right)}{\sum_{j'=1}^{n} exp\left(\frac{Q_i K_{j'}^\top}{\sqrt{d_{out}}}\right)} V_j \in \mathbb{R}^{d_{\mathrm{out}}}$$

$$= \sum_{j=1}^{n} \frac{k(Q_i, K_j)}{\sum_{j'=1}^{n} k(Q_i, K_j)} V_j \in \mathbb{R}^{d_{\mathrm{out}}},$$

with k a non-negative kernel function, which can be seen as a kernel smoothing.

Kernel construction.

- Different choices for k suggest different transformers architectures [Tsai et al., 2019].

Self-attention for large (biological) sequences

Dealing with small datasets of large sequences.

- Sequence: a set of features with 1-D positional information.
- Important applications, *e.g*, protein sequences in bioinformatics.
- Long-range and potentially complex dependencies between elements.
- Varying size of the sequences.

Biological sequences bring two more problems.

- Long sequences (1000+ base pairs).
- Few labeled data (e.g, 20 samples per class for SCOP1.75).

```
LDKVEAEVQIDRLITG
```

Figure 1: Short part of mRNA sequence for the SARS-Cov-2 spike protein (each symbol represents an amino-acid).

Self-attention for large (biological) sequences

Transformers are delicate to use in this setting.

- Attractive inductive bias.
- Small amount of data.
- Memory issues for large sequences (although recently alleviated by the *efficient transformers* line of work, see [Tay et al., 2020]).

We propose a self-attention like embedding for sequences [Mialon et al., 2021a].

- Our embedding will provide a natural notion of pooling.
- The attention weights will be the output of a matching operation.
- We choose optimal transport, as it benefits from a rich theory and efficient solvers.

Optimal Transport

Distributing mass with minimal cost.

- Let a in Δ^n (probability simplex) and b in $\Delta^{n'}$ be weights of the discrete measures $\sum_i \mathrm{a}_i \delta_{\mathrm{x}_i}$ and $\sum_j \mathrm{b}_j \delta_{\mathrm{x}'_j}$ with respective locations x and x', where δ_{x} is the Dirac at position x.
- Let C in $\mathbb{R}^{n \times n'}$ be a pairwise cost matrix.
- The entropic regularized Kantorovich relaxation of OT from x to x' is

$$\min_{\mathrm{P} \in U(\mathrm{a},\mathrm{b})} \sum_{ij} \mathrm{C}_{ij}\mathrm{P}_{ij} - \varepsilon H(\mathrm{P}), \tag{2}$$

with $H(\mathrm{P}) = -\sum_{ij} \mathrm{P}_{ij}(\log(\mathrm{P}_{ij}) - 1)$ is the entropic regularization with parameter ε (controls sparsity of P), and U is the space of admissible couplings between a and b:

$$U(\mathrm{a},\mathrm{b}) = \{\mathrm{P} \in \mathbb{R}_+^{n \times n'} : \mathrm{P}1_n = \mathrm{a} \text{ and } \mathrm{P}^\top 1_{n'} = \mathrm{b}\}.$$

- Typically solved using Sinkhorn's algorithm [Sinkhorn and Knopp, 1967, Cuturi, 2013].
- In practice, a and b will be uniform measures.

Constructing a self-attention embedding

Optimal Transport Embedding and Kernel. Let $X \in \mathbb{R}^{n \times d}$, a sequence of features. κ a p.d. kernel with associated embedding φ.

- We define our embedding $\Phi_Z \in \mathbb{R}^{p \times d}$ as

$$\Phi_Z(X) = \sqrt{p} \times \mathrm{P}_\kappa(X, Z)^\top \varphi(X).$$

- $\mathrm{P}_\kappa(X, Z)$ the OT plan between X with cost $-\kappa$ and a learned reference $Z \in \mathbb{R}^{p \times d}$ item $\varphi(X) := [\varphi(\mathrm{X}_1), \dots, \varphi(X_n)]^\top$, with $\varphi : \mathbb{R}^d \to \mathcal{H}$
- Its associated p.d. kernel is

$$K_Z(X, X') = \sum_{i,j} \mathrm{P}_{\kappa,Z}(X, X')_{ij} \kappa(X_i, X'_j),$$

with $\mathrm{P}_{\kappa,Z}(X, X') := p \times \mathrm{P}_\kappa(X, Z)\mathrm{P}_\kappa(X', Z)^\top$.

Constructing a self-attention embedding

Kernel interpretation of our embedding.

- $\mathsf{P}_{\kappa,Z}(X, X')$: valid transport plan [Peyré and Cuturi, 2019], rough approximation of $\mathsf{P}_{\kappa}(X, X')$.
- K_Z is a p.d. surrogate for $K_{\mathrm{OT}}(X, X') = \sum_{i,j} \mathsf{P}_{\kappa}(X, X')_{ij} \kappa(X_i, X'_j)$.
- K_{OT} induces the 2-wasserstein distance and is not p.d. [Rubner et al., 2000].

Getting back to the kernel smoothing formula.

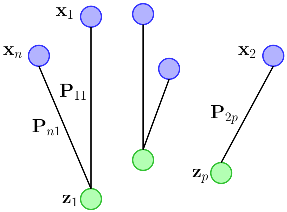
- We replaced $\frac{k(Q_i, K_j)}{\sum_{j'=1}^{n} k(Q_i, K_{j'})}$ by $\mathsf{P}_{\kappa}(X, Z)_{ij}$.

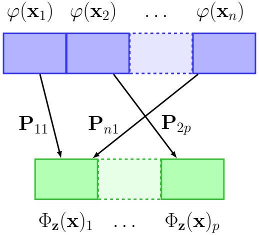
Result: a pooled self-attention embedding

- We introduced

$$\Phi_Z(X) = \sqrt{p} \times \mathrm{P}_\kappa(X, Z)^\top \varphi(X),$$

  which simultaneously embeds and pools elements of an input sequence.
- Non-linear embedding via φ.
- Pooling via P_κ, similar elements are pooled together.
- Natural notion of pooling by choosing $p < n$.





Result: a pooled self-attention embedding

Learning our embedding.

- Without supervision: simple k-means for Z and a tractable approximation of φ [Mairal, 2016].
- With supervision: back-propagating through a few steps of Sinkhorn iterations for Z. Classical back-propagation for a tractable approximation of φ [Mairal, 2016].

Extensions.

- Relative position encoding.
- Multi-head.

Experiments

SCOP1.75: Protein fold classification. $\sim$ 20000 samples, $\sim$ 1000 labels, many sequences longer than 1000 base pairs.

Table 1: Classification accuracy (top 1/5/10) on test set for SCOP 1.75 for different unsupervised and supervised baselines, averaged from 10 different runs. (q references $\times$ p supports).

Method	Unsupervised	Supervised
DeepSF [Hou et al., 2019]	Not available.	73.0/90.3/94.5
CKN [Chen et al., 2019a]	81.8±0.8/92.8±0.2/95.0±0.2	84.1±0.1/94.3±0.2/96.4±0.1
RKN [Chen et al., 2019b]	Not available.	85.3±0.3/95.0±0.2/96.5±0.1
Set Transformer [Lee et al., 2019]	Not available.	79.2±4.6/91.5±1.4/94.3±0.6
Approximate Rep the Set [Skianis et al., 2020]	Not available.	84.5±0.6/94.0±0.4/95.7±0.4
Ours (dot-product instead of OT)	78.2±1.9/93.1±0.7/96.0±0.4	87.5±0.3/95.5±0.2/96.9±0.1
Ours (Unsup.: 1 × 100 / Sup.: 5 × 10)	**85.8±0.2/95.3±0.1/96.8±0.1**	**88.7±0.3/95.9±0.2/97.3±0.1**

Discussion

Connection to transformers

- Relationship to efficient transformers [Kitaev et al., 2020].
- Kernel methods vs. few-shot learning with pre-trained models for biological sequences [Rives et al., 2019].

Code

- Freely available at `https://github.com/claying/OTK`.

Learning on graph data

Graph data are very valuable.

- Proteins in computational biology [Senior et al., 2020].
- Molecules in chemoinformatics [Duvenaud et al., 2015].
- Shapes in computer vision and computer graphics [Verma et al., 2018], etc.

Graph Neural Networks (GNNs).

- Originally introduced as an extension of convolutions for graph-structured data [Scarselli et al., 2008].
- Message passing paradigm in which vectors (messages) are exchanged (passed) between neighboring nodes whose representations are updated using neural networks.
- Many strategies to aggregate features of neighboring nodes [Bronstein et al., 2017, Duvenaud et al., 2015].
- De facto architecture for graph structured data.

Challenging GNNs with Structural Graph Transformers

GNNs and transformers are tightly connected, but...

- GNNs are the standard architecture for learning on graphs. Inductive bias: message passing between neighbors.
- Transformers: all input elements are allowed to communicate.
- Self-attention layer is permutation invariant, hence the need for structure encoding.

How to provide the transformer with graph structural information?

- Structural Graph Transformers [Mialon et al., 2021b]

Two mechanisms for providing transformers with graph structural information

Relative node position encoding.

- Position encoding: adding positional only information to the feature vector of an input node or to the attentions scores.
- As opposed to sequences or images, encoding positions of the elements in a graph is not trivial.
- [Dwivedi and Bresson, 2021] proposed absolute position encoding strategy based on the eigenvectors of the Laplacian. Blind spot with respect to transferability between graphs.

Leveraging substructures.

- Substructures: carry local positional information and content, *e.g* walks, subtrees, graphlets.
- Heavily used within graph kernels [Borgwardt et al., 2020].

First mechanism: Relative position encoding with kernels on graphs

Spectral graph analysis.

- The Laplacian of a graph with n nodes defined as $L = D - A$. D is a $n \times n$ diagonal matrix of node degrees and A the adjacency matrix.
- Eigenvalue decomposition $L = \sum_i \lambda_i u_i u_i^\top$.
- The eigenvalue $\lambda_i = u_i^\top L u_i$ characterizes the amount of oscillation of the corresponding eigenvector u_i (a function on the nodes).
- For this reason, this decomposition is viewed as the discrete equivalent to the sine/cosine Fourier basis in $\mathbb{R}^n$ and associated frequencies.

Remark. Very often, the normalized Laplacian $I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$ is used instead of L, which does not change the above interpretation.

First mechanism: Relative position encoding with kernels on graphs

Kernels on graphs.

- It is possible to define a family of p.d. kernels on the graph [Smola and Kondor, 2003] by applying a regularization function r to the spectrum of L.
- We get a rich class of kernels

$$K_r = \sum_{i=1}^{m} r(\lambda_i) u_i u_i^\top, \tag{3}$$

  associated with the norm $\|f\|_r^2 = \sum_{i=1}^{m} (f_i^\top u_i)^2 / r(\lambda_i)$ from a reproducing kernel Hilbert space (RKHS), where $r : \mathbb{R} \mapsto \mathbb{R}_*^+$ is a non-increasing function such that smoother functions on the graph would have smaller norms in the RKHS.

First mechanism: Relative position encoding with kernels on graphs

Diffusion Kernel [Kondor and Vert, 2004].

- When $r(\lambda_i) = e^{-\beta\lambda_i}$,

$$K_D = \sum_{i=1}^{m} e^{-\beta\lambda_i} u_i u_i^\top = e^{-\beta L} = \lim_{p \to +\infty} \left(I - \frac{\beta}{p} L \right)^p.$$

- Discrete equivalent of the Gaussian kernel, a solution of the heat equation in the continuous setting, hence its name.
- Interpretation in terms of diffusion of a substance in the graph, controlled by β.

First mechanism: Relative position encoding with kernels on graphs

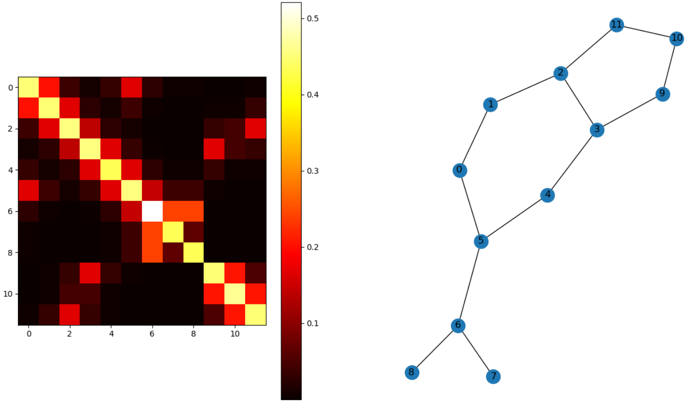


Figure 3: Diffusion kernel between the nodes of a MUTAG sample graph ($\beta = 1$).

First mechanism: Relative position encoding with kernels on graphs

Modulating the attention scores.

- Self-attention layer becomes

$$\text{PosAttention}(Q, V, K_r) = \text{normalize}\left(\exp\left(\frac{QQ^\top}{\sqrt{d_{\text{out}}}}\right) \odot K_r\right) V \in \mathbb{R}^{n \times d_{\text{out}}}, \tag{4}$$

  with the same Q and V matrices, and K_r a kernel on the graph.

- During forward pass, feature map X is updated as follows:

$$X = X + D^{-\frac{1}{2}} \text{PosAttention}(Q, V, K_r), \tag{5}$$

  with D the matrix of node degrees and K_r a kernel on the graph.

Remark. As opposed to absolute position encoding, the model does not rely on the transferability of eigenvectors between different Laplacians.

First mechanism: Relative position encoding with kernels on graphs

Back to the kernel smoothing formula.

- We replaced $k(Q_i, K_j)$ by $k(Q_i, K_j) \times K_r(X_i, X_j)$.
- As observed in [Tsai et al., 2019] for sequences, this is an approach related to relative positional encoding [Shaw et al., 2018].

Second mechanism: Leveraging substructures via kernel embedding of paths

Graph convolutionnal kernel networks (GCKN) [Chen et al., 2020].

- Let us consider a graph G with n nodes, $\mathcal{P}_k(u)$ the set of paths shorter than or equal to k that start with node u, and p in $\mathcal{P}_k(u)$ will denote the concatenation of all node features encountered along the path.
- A layer of GCKN defines a feature map X in $\mathbb{R}^{n \times d}$ such that

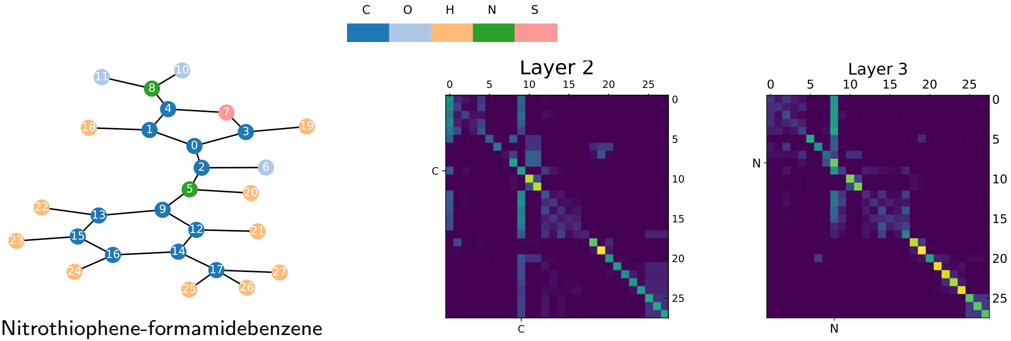
  $$X(u) = \sum_{p \in \mathcal{P}_k(u)} \psi(p),$$

  with $X(u)$ the column of X corresponding to node u and ψ is a d-dimensional embedding of the path features p.
- We encode a node as the sum of its features and those produced by one GCKN layer.

SGT is able to outperform popular GNNs

Method / Dataset	MUTAG	PROTEINS	PTC	NCI1	ZINC (no edge feat.)
Size	188	1113	344	4110	12k
Max. number of nodes	28	620	109	111	37
GCN [Kipf and Welling, 2017]	78.9±10.1	75.8±5.5	54.0±6.3	75.9±1.6	0.367±0.011
GAT [Veličković et al., 2018]	80.3±8.5	74.8±4.1	55.0±6.0	76.8±2.1	0.384±0.007
GIN [Xu et al., 2019]	82.6±6.2	73.1±4.6	55.0±8.7	**81.7±1.7**	0.387±0.015
[Dwivedi and Bresson, 2021]	83.9±6.5	70.1±3.2	57.7±3.1	80.0±1.9	0.323±0.013
Transformers (T)	82.2±6.3	75.6±4.9	58.1±10.5	70.0±4.5	0.696±0.007
T + LapPE	85.8±5.9	74.6±2.7	55.6±5.0	74.6±1.9	0.507±0.003
T + Adj PE	87.2±9.8	72.4±4.9	59.9±5.9	79.7±2.0	0.243±0.005
T + 2-step RW kernel	85.3±6.9	72.8±4.5	**62.0±9.4**	78.0±1.5	0.243±0.010
T + 3-step RW kernel	83.3±6.3	**76.2±4.4**	61.0±6.2	77.6±3.6	0.244±0.011
T + Diffusion kernel	82.7±7.6	74.6±4.2	59.1±7.4	78.9±1.6	0.255±0.010
T + GCKN	84.4±7.8	69.5±3.8	61.5±5.8	78.1±5.1	0.274±0.011
T + GCKN + 2-step RW kernel	90.4±5.8	72.5±4.6	58.4±7.6	81.0±1.8	0.213±0.016
T + GCKN + Adj PE	**90.5±7.0**	71.1±6.9	57.9±4.2	81.4±2.2	**0.211±0.010**

Patterns captured in the attention scores of SGT



Nitrothiophene-formamidebenzene

Figure 4: A molecule from the Mutagenicity data set [Kersting et al., 2016]. The attention scores are averaged by heads. *Left*: node 9 (C of aromatic cycle) is salient. *Right*: nodes 8 (N of NO_2) and 17 (C of CH_3) are salient. NO_2 is known for its mutagenetic properties.

Conclusions

Kernel methods

- Reconciles deep learning with small data regimes.
- Understanding architectures via a new lens.

Optimal Transport Embedding

- Dealing with long sequences with few data.
- Connection to the recent line of work efficient transformers.
- Challenged by few-shot learning with pre-trained models.

Structural Graph Transformers

- Inductive bias of transformers is valid with graph.
- Attention provides promising intepretation tools.

References I

Borgwardt, K., Ghisu, E., Llinares-López, F., O'Bray, L., and Rieck, B. (2020). Graph kernels: State-of-the-art and future challenges. *arXiv preprint arXiv:2011.03854*.

Bronstein, M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. (2017). Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42.

Chen, D., Jacob, L., and Mairal, J. (2019a). Biological sequence modeling with convolutional kernel networks. *Bioinformatics*, pages 35(18):3294–3302.

Chen, D., Jacob, L., and Mairal, J. (2019b). Recurrent kernel networks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Chen, D., Jacob, L., and Mairal, J. (2020). Convolutional kernel networks for graph-structured data. In *International Conference on Machine Learning (ICML)*.

Cuturi, M. (2013). Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems (NeurIPS)*.

References II

Duvenaud, D., Maclaurin, D., Aguilera-Iparraguirre, J., Gómez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A., and Adams, R. P. (2015). Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Dwivedi, V. P. and Bresson, X. (2021). A generalization of transformer networks to graphs. *AAAI Workshop on Deep Learning on Graphs: Methods and Applications*.

Hou, J., Adhikari, B., and Cheng, J. (2019). Deepsf: deep convolutional neural network for mappingprotein sequences to folds. *Bioinformatics*, pages 34(8):1295–1303.

Kersting, K., Kriege, N. M., Morris, C., Mutzel, P., and Neumann, M. (2016). Benchmark data sets for graph kernels.

Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Kitaev, N., Łukasz Kaiser, and Levskaya, A. (2020). Reformer: The efficient transformer. In *International Conference on Learning Representations (ICLR)*.

References III

Kondor, R. and Vert, J.-P. (2004). Diffusion kernels. In *Kernel Methods in Computational Biology*, pages 171–192. MIT Press.

Lee, J., Lee, Y., Kim, J., Kosiorek, A. R., Choi, S., and Teh, Y. W. (2019). Set transformer: A framework for attention-based permutation invariant neural networks. In *International Conference on Machine Learning (ICML)*.

Mairal, J. (2016). End-to-end kernel learning with supervised convolutional kernel networks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Mialon, G., Chen, D., d'Aspremont, A., and Mairal, J. (2021a). A trainable optimal transport embedding for feature aggregation and its relationship to attention. In *International Conference on Learning Representations (ICLR)*.

Mialon, G., Chen, D., Selosse, M., and Mairal, J. (2021b). Structural graph transformers.

Peyré, G. and Cuturi, M. (2019). Computational optimal transport. *Foundations and Trends in Machine Learning*, 11(5-6):355–206.

References IV

Rives, A., Goyal, S., Meier, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., and Fergus, R. (2019). Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. In *bioRxiv 622803*.

Rubner, Y., Tomasi, C., and Guibad, L. J. (2000). The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40:99–121.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2008). The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80.

Senior, A. W., Evans, R., Jumper, J., Kirkpatrick, J., Sifre, L., Green, T., et al. (2020). Improved protein structure prediction using potentials from deep learning. *Nature*, 577(7792):706–710.

Shaw, P., Uszkoreit, J., and Vaswani, A. (2018). Self-attention with relative position representations. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Sinkhorn, R. and Knopp, P. (1967). Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2).

References V

Skianis, K., Nikolentzos, G., Limnios, S., and Vazirgiannis, M. (2020). Rep the set: Neural networks for learning set representations. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Smola, A. J. and Kondor, R. (2003). Kernels and regularization on graphs. In Schölkopf, B. and Warmuth, M. K., editors, *Learning Theory and Kernel Machines*, pages 144–158. Springer Berlin Heidelberg.

Tay, Y., Dehghani, M., Bahri, D., and Metzler, D. (2020). Efficient transformers: A survey.

Tsai, Y.-H. H., Bai, S., Yamada, M., Morency, L.-P., and Salakhutdinov, R. (2019). Transformer dissection: A unified understanding of transformer's attention via the lens of kernel. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018). Graph attention networks. In *International Conference on Learning Representations (ICLR)*.

References VI

Verma, N., Boyer, E., and Verbeek, J. (2018). Feastnet: Feature-steered graph convolutions for 3d shape analysis. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2019). How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*.