Towards Generative Drug Discovery: Metric Learning using Variational Autoencoders

Andrew Gritsevskiy

MIT PRIMES

September 2017

Abstract

We report a method for metric learning using an extended variational autoencoder. Our architecture, based on deep learning, provides the ability to learn a transformation-invariant metric on any set of data.

Our architecture consists of a pair of encoding and decoding networks. The encoder network converts the data into differentiable latent representations, while the decoder network learns to convert these representations back into data. We then apply an additional set of losses to the encoder network, forcing it to learn codings that are independent of orientation and reflect the desired metric. Then, our architecture is able to predict the real metric for a set of data points, and can generate data points that match a set of requirements.

We demonstrate our networks ability to calculate the maximum overlap area of any two shapes in one shot; we also demonstrate our networks success at matching halves of geometric shapes. We then propose the applications of our network to areas of biochemistry and medicine, especially generative drug discovery.

Introduction

The fundamental goal of drug design is to develop molecules that fit certain properties. Given that the size of the chemical drug space is estimated to be 10^{60},[1] it is reasonable to expect that there are subsets of this space that match any reasonable selection of biochemical requirements. Since current drug screening methods often depend on automated testing of millions of molecules in laboratories, only about 10^6 molecules have ever been synthesized.[2] Attempts to improve those numbers with computational methods generally focus on virtual screening, which still requires advanced simulations and great computational power; as a result, the number of chemical structures ever considered for drugs has been put at 10^8.[2] In this paper, we propose a step towards generative drug discovery by applying metric learning to differentiable representations of molecules. We then present experimental verification on a toy task of shape matching, propose applications of our architecture to drug design, and discuss further uses of our system in other fields.

The main design of our architecture is based on the idea of an autoencoder, a neural network used for learning features without supervision. In this paper, we propose a network based on an extended variational autoencoder, which we call a magic autoencoder, for implementing metric learning.

Related work

When developing this system, we were greatly inspired by advances in Bayesian inference and deep learning,[3] as well as recent successes of autoencoders in providing data-driven, continuous representations of molecules.[4] Further, we looked at existing molecular representations through molecular convolutions,[5] and abstract metricized representations of concepts, such as those described in the word2vec model by Mikolov et al.[6][7]

Model

To describe our model, we discuss the uses of variational autoencoders (abbreviated VAEs) on transforming data into a differentiable representation. VAEs consist of two networks—an encoder network and a decoder network, both often recurrent neural networks.[8] The encoder network learns to represent the input data in a latent space of arbitrary dimensionality but limited hidden unit size, thus extracting important features from the data. The decoder network learns to symmetrically convert this latent representation back into data space. VAEs employ two loss terms, the reconstruction loss and the latent loss, for optimizing reconstruction quality while maintaining a reasonable exploration of latent space.[3] If we sample the latent space on a regular interval, we can see that VAEs generally learn a continuous representation of the data, as demonstrated for handwritten digits in Figure 1.1. However, the arrangement of the latent space is not necessarily constrained in any meaningful way. Thus, we add two loss terms to a VAE-like structure, one term forcing the encoder network to learn transformation invariant representations of the input data, and the other morphing the latent space into one with a meaningfully defined metric. These losses are then minimized with Adam, a stochastic optimization algorithm introduced by Kingma et al.[9]

Network architecture

The way our network learns is based on pairs of data objects with some defined metric between them, such as binding affinity; the actual input to our network consists of four pieces of information. First, the network takes two vectors representing the input object and a randomly rotated and translated version of the input object—for instance, an EM map of a molecule and an EM map of the same molecule in a random spatial orientation. Next, the network is given the second input object, such as an EM map of another molecule. Finally, the network is given the value of the metric between these two objects, such as the binding affinity. The three vectors are all fed into the encoder network, producing latent

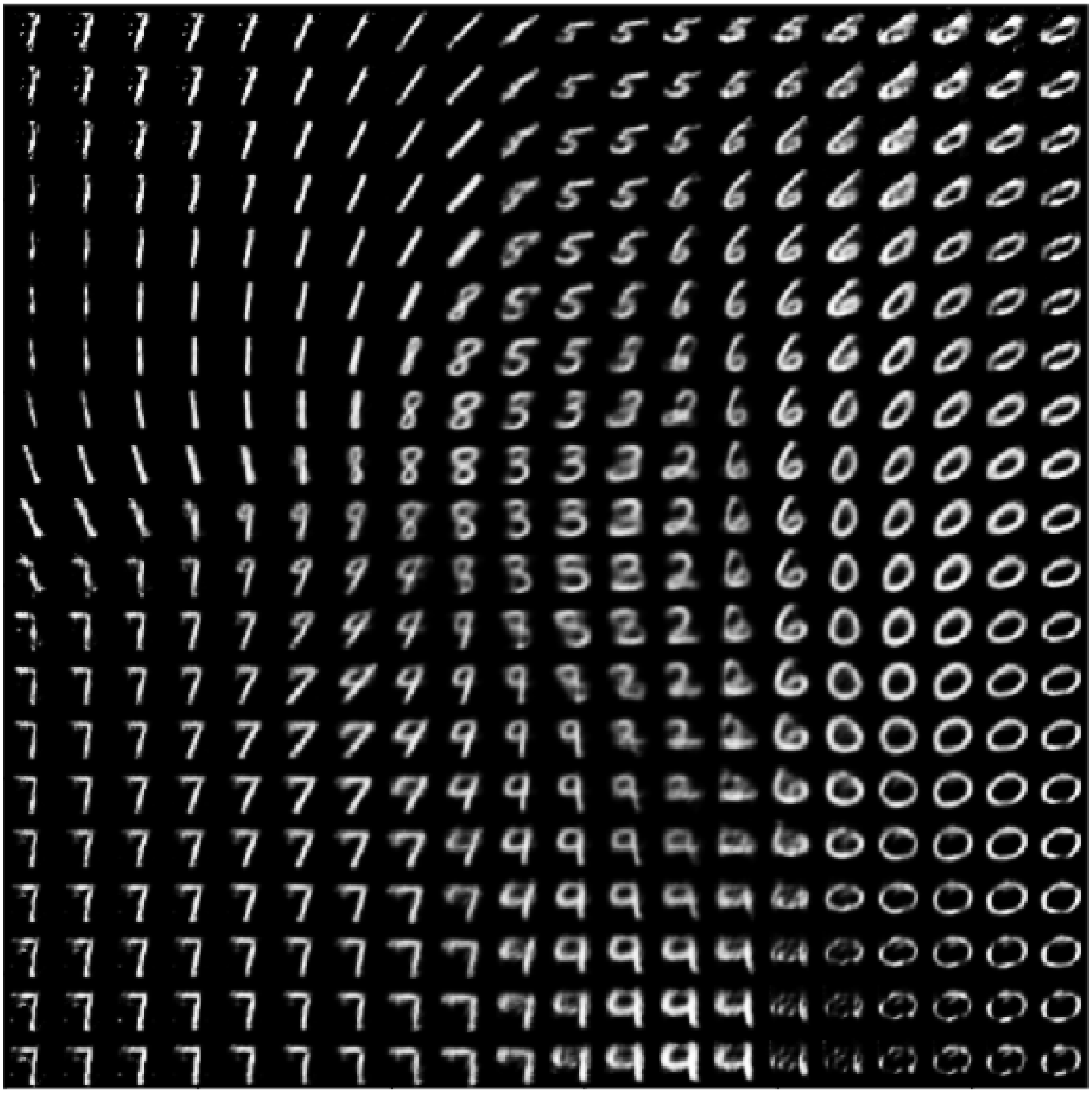

Figure 1.1: a variational autoencoder trained on handwritten digits generates a continuous representation of the data. Here, the dimensionality of the latent space was restricted to 2 and mapped on a plane. However, for actual data, the dimensionality of the latent space can easily span hundreds or even thousands of dimensions.

representations of the three vectors, which well call z, z_r, and z_b, respectively. Now, the loss term for the autoencoder is defined as follows:

$$\mathcal{D}_{KL}(q_\phi(z|x)||p_\phi(z)) - \mathbb{E}_{z\sim q_\phi}[log(p_\phi(x|z))] + \frac{10}{n}\sum_{i=1}^{n}(z - z_r)^2 \tag{1}$$

This loss term is essentially the same as that for a VAE; however, there is an added term—namely, the squared difference between the latent representation of the first data point and the latent representation of the rotated first data point. This incentivises the encoder to learn a transformation-independent coding, which is one of the crucial steps for structure-based one-shot learning. The metricization loss is defined as

$$\frac{\|\overrightarrow{z}\|\,\|\overrightarrow{z_b}\|}{\overrightarrow{z}\cdot\overrightarrow{z_b}} - m \tag{2}$$

where m is the value of the metric between the first object and the second object. That is, we penalize any latent representations of the two objects where the inverse of the cosine similarity between them is not equal to the metric, based on how far our approximation is from the actual value. These two losses combined should be able to learn the following: first, a translation and rotation-invariant encoding of the input data; second, a meaningful metric over the differentiable and compact latent space; and finally, a decoder from the latent space back to data space. Generally, we train the two loss terms with separate optimizers in order to have finer control over the learning rates; however, this is not strictly necessary. Figure 1.2 contains a complete representation of the model architecture.

Experiments

Implementation

All of our experiments were implemented in Tensorflow and tested on an Intel(R) Xeon(R) E5-2620 v4 CPU and accelerated with one Nvidia GPU, unless otherwise specified. An implementation of our architecture can be found at [REDACTED].

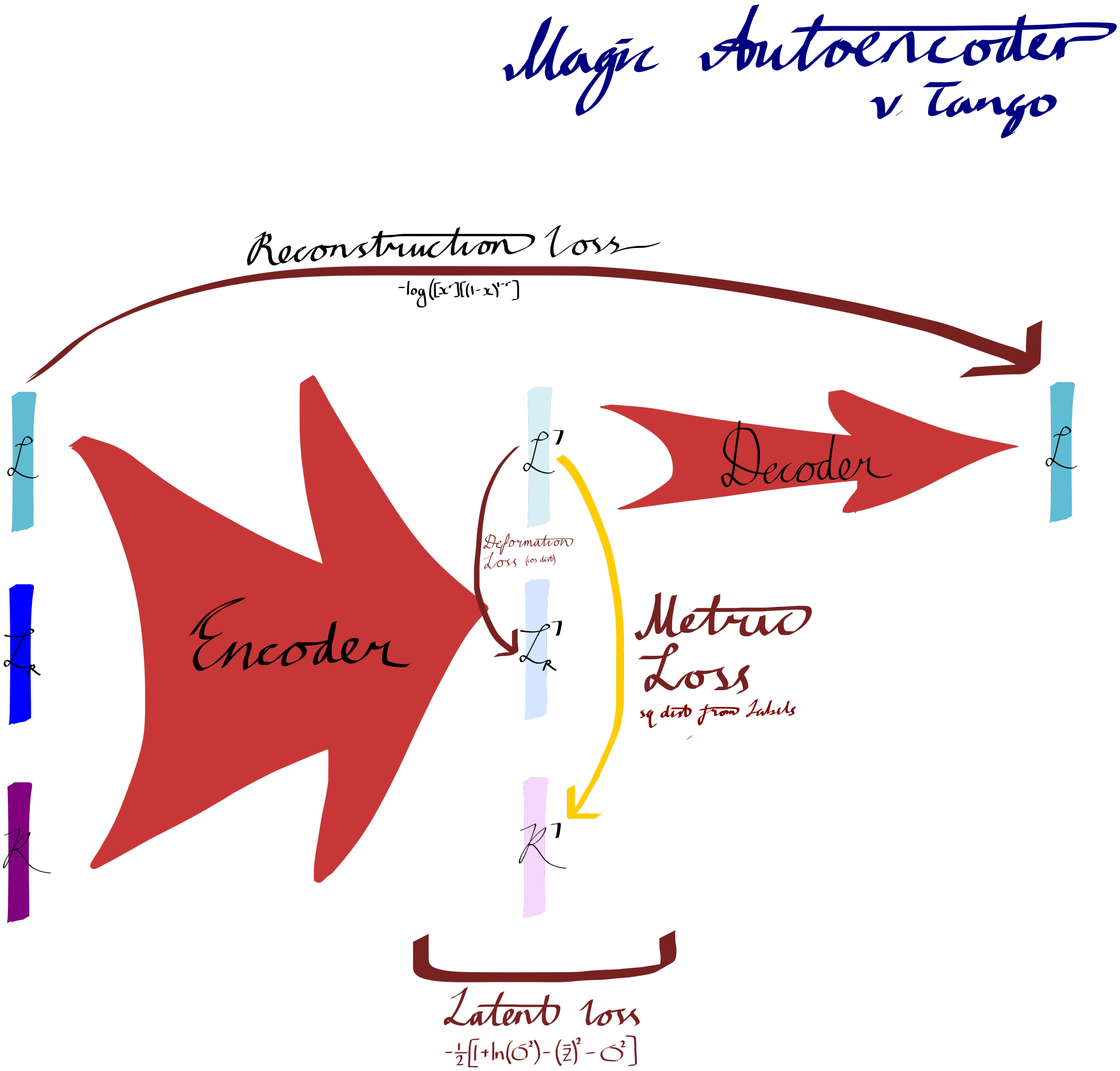

Figure 1.2: the magic autoencoder. The losses are the reconstruction loss, which makes the autoencoder learn proper encoding and decoding functions; the latent loss, which lets the autoencoder reasonably arrange the latent space; the deformation loss, which enforces transformation invariance over the learned representations; and the metric loss, which implements the metric learning itself.

Shape overlap

The main dataset we used for testing this network is the shape overlap dataset.[10] Calculating the maximum overlap area between two shapes is an extremely complex problem, somewhat representative of drug docking in two dimensions. While it is simple to come up with an algorithm that samples thousands of configurations of the two shapes and returns the maximum discovered overlap area, such an approach would take thousands of CPU-hours for just dozens of figures. In order to mitigate the wild inefficiency of this approach, evolutionary algorithms may be applied, providing a significant boost in speed, and even potential increases in accuracy. However, we would like to predict the answer in one shot—that is, given two shapes, we would like a network that predicts the maximum overlap area between them without necessarily calculating the optimal position. The shape overlap dataset provides 100,000 pairs of shapes, together with the maximum overlap area, as shown in Figure 1.3.

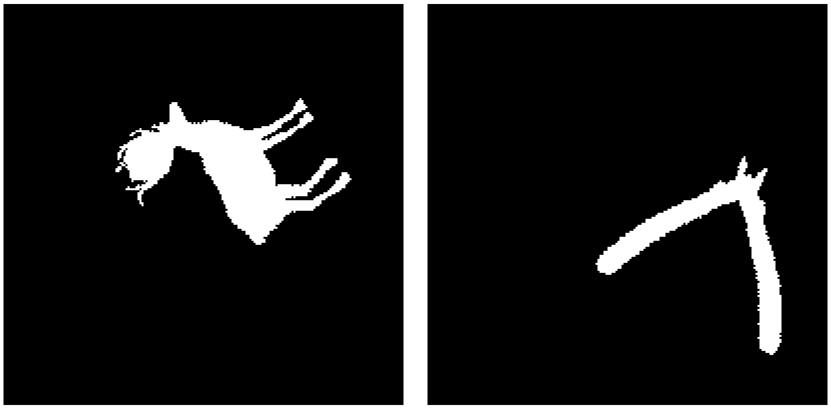

Figure 1.3: an example from the shape overlap dataset. The maximum overlap area between the shape on the left and the shape on the right is equal to 1783 pixels, which our network predicts.

Our network is set up according to our architecture. The first input is a vector representing the first image, the second input is a randomly transformed variation of the first image, the third input is the second image, and the last input is the maximum overlap area between the two images. These inputs are fed into our network, with the two optimizers' respective learning rates being 1e-8 and 1e-10. Our fully-connected encoder network contains 40000, 500, 500, and 500 neurons in its four layers, while the two-layer fully-connected decoder network has 500 and 500. If we set the number of latent dimensions to 2 and plot the resulting images, we can see in Figure 1.4 that the autoencoder does indeed learn a continuous representation of the input. However, as a result of experimental verification, we have determined that a latent space dimensionality of 80 tends to work best.

Shape competion

Another method of experimental verification we used on our network was testing it on the MSHAPES shape completion task.[11] The dataset contains 50,000 pairs of geometric shapes cut into halves. The three inputs for the autoencoder become, respectively, the first half of the shape, the rotated first half of the shape, and the second half of the shape. Since this is a binary classification task, we assign an arbitrary threshold for the metric–in our case, we said that a cosine distance between two encoded vectors of greater than 1.0 meant that the halves do not match, while a cosine distance of 1.0 or less meant that the halves match together to form a shape. Our network excelled at this task, performing significantly better than random, as discussed in the next section. We note that matching shapes is recognized as an easier task than calculating the maximum overlap area; however, the networks excellent performance at this task underlines its generalizability and extensibility to different areas of neural predictions, such as binary classification.

Results

Shape overlap prediction

To experimentally verify our network, we compared the mean-squared error in predicting the areas between one batch of 24 pairs of shapes and the error in guessing the overlap areas in the batch. The experimental results are shown in Table 1.

Using a two-tailed z-test where the null hypothesis is that the error of the autoencoder

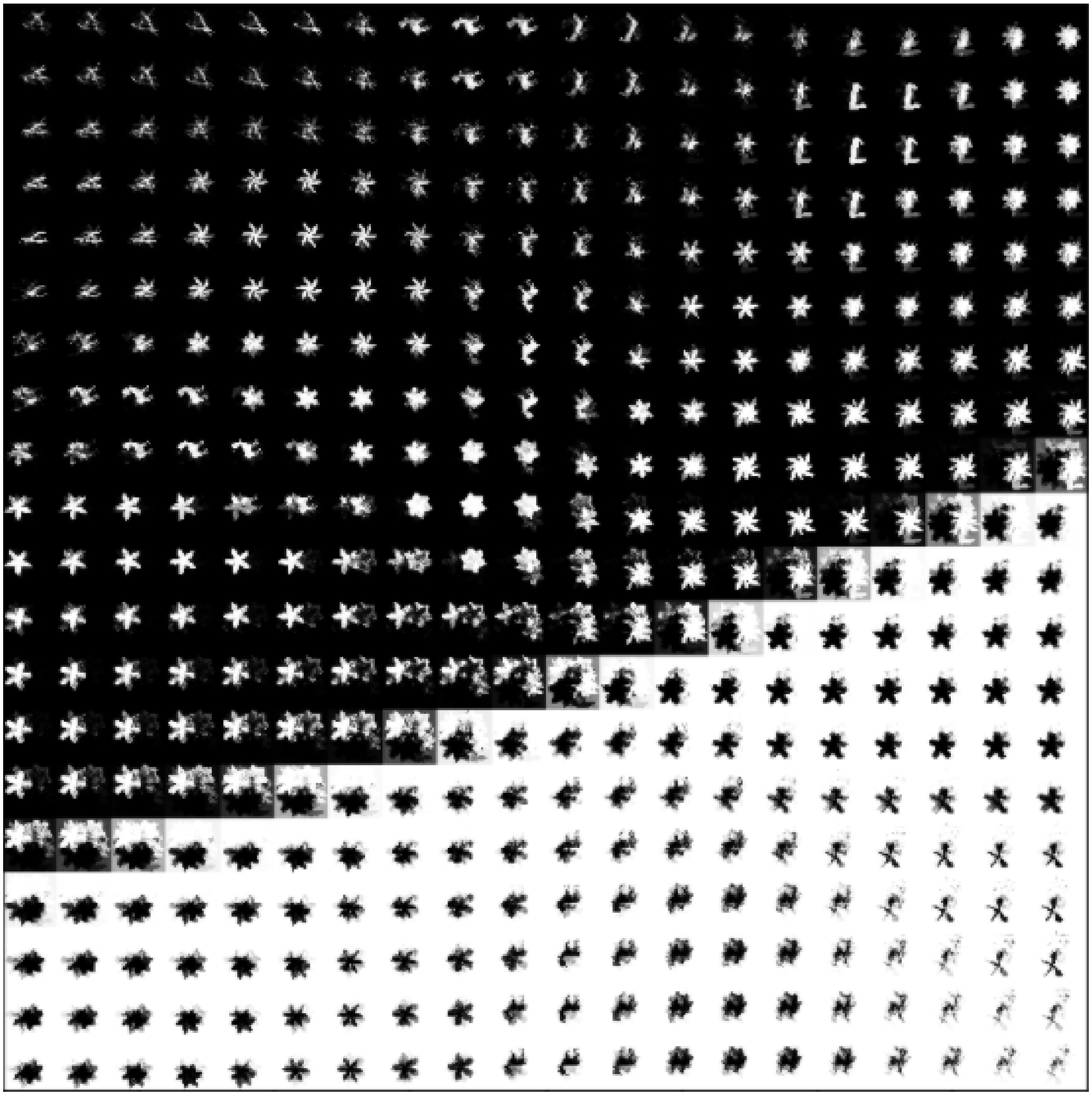

Figure 1.4: a sampling of the 2d latent space for the shape matching tasks. Here, only the first, non-rotated images are displayed, so distances are meaningless, since the metric is not defined. If we look across the rows and columns, we can see a that the network learned a continuous latent representation of the shapes, as demonstrated by the relatively smooth transition from depictions of wrenches to stars, to scissors to moose.

Table 1: Comparison of naïve error and autoencoder error

Trial	Naïve method error	Magic autoencoder error
1	6.39e6	4.78e6
2	5.06e6	4.22e6
3	5.75e6	2.75e6
4	7.36e6	5.88e6
5	7.12e6	4.32e6
6	5.64e6	6.46e6

is identical to the naive methods error, and the alternate hypothesis is that the error of the magic autoencoder is different than that of the naive method, we obtain a p-value of 0.049165. Since $p<0.05$, we conclude that our architecture is significantly better at predicting the overlap area of random shapes than current methods. Thus, this lends credence to the idea that this neural architecture is a powerful way to predict any reasonably defined metric over arbitrary data; in this case, maximum overlap area.

Shape completion prediction

Our network performed extremely well on the shape completion task, getting near-perfect accuracy after several hours of training on a Intel(R) Xeon(R) E5-2620 v4 CPU with one Nvidia GPU. Table 2 compares the mean batchwise classification error between randomly guessing the class and our networks prediction, while Figure 1.5 shows the decreasing error of the magic autoencoders predictions as the network is trained.

Table 2: Comparison of Gaussian noise-based guessing error and magic autoencoder error

Trial	Gaussian noise error	Magic autoencoder error
1	1.416666667	0.286658
2	1.583333333	0.258178
3	1.416666667	0.240011
4	1.333333333	0.212152
5	1.25	0.282565
6	1.416666667	0.281238

Utilizing a two-tailed z-test where the null hypothesis is that the error of the autoencoder is identical to the naive methods error, and the alternate hypothesis is that the error of the magic autoencoder is different than random, we obtain a p-value of less than 1e-7. Since p<0.0000001, we conclude that our architecture is significantly better at connecting shapes than random guessing. This does not come as a surprise, given our networks great performance at the much harder task detailed above. However, the extendability of our network to many classes of problems, such as binary classification, demonstrates that our architecture is generalizable to any task, even with a discrete or binary metric.

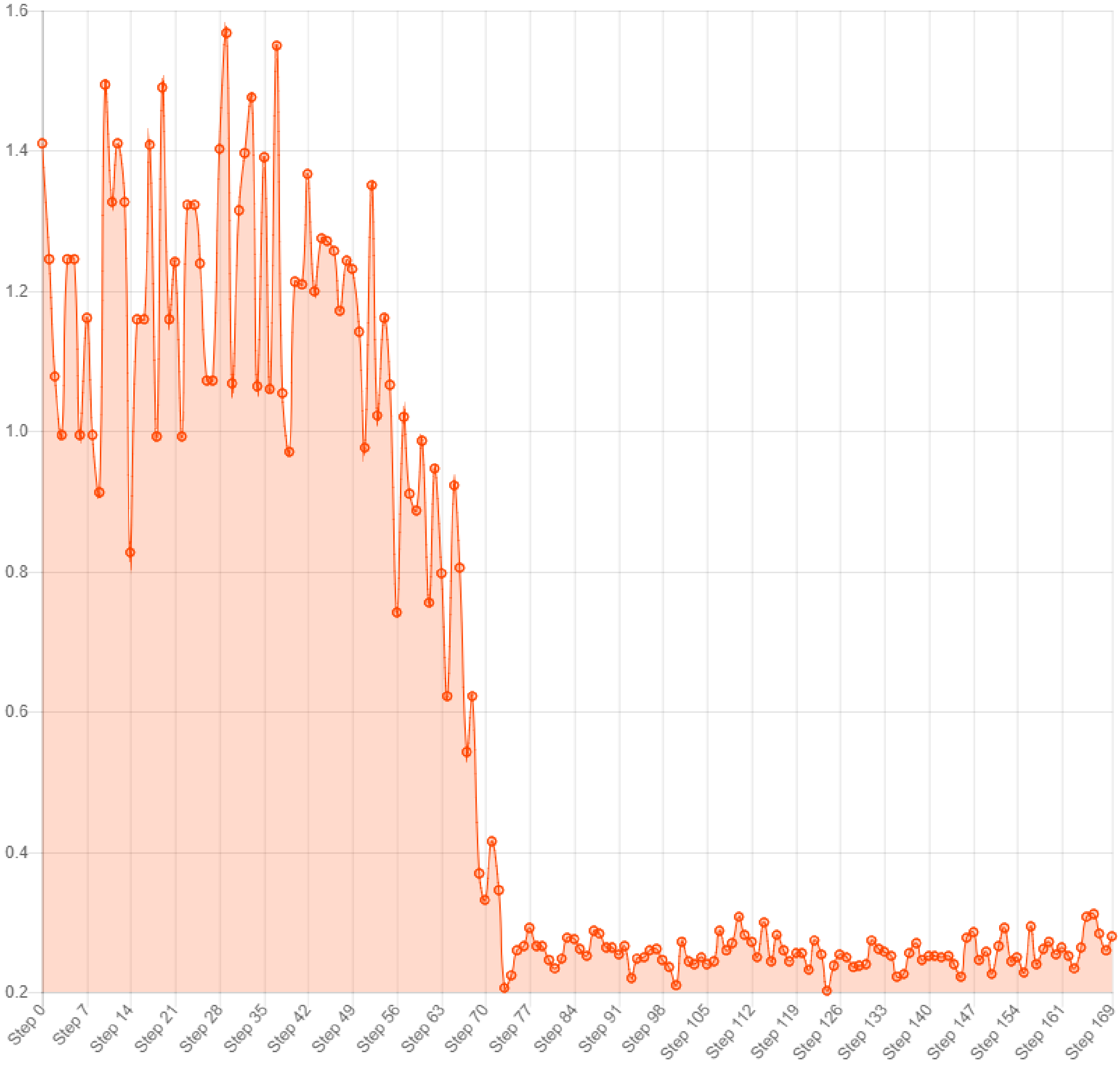

Figure 1.5: the metric training loss during experimental verification of the magic autoencoder for the shape matching dataset.

Discussion

Based on the experimental verification we conducted, we can conclude that our architecture is well-suited for transformation-invariantly predicting both continuous and discrete metrics over arbitrary sets of data. The magnitude of improvement between the shape competition result and the overlap prediction result can be explained by the fact that overlap prediction is a significantly more difficult problem; however, our network's performance suggests that we can apply it to other complex tasks as well. The main task that we are applying our network to now is metric learning over molecular data. As mentioned earlier, using EM map representations of molecules with a metric of binding affinity has produced very encouraging preliminary results. We plan on using this to improve over suites such as Autodock Vina and Glide both in terms of accuracy and precision, as our network seems to beat both programs' margins of error on two-dimensional tasks.

Finally, our network can be easily modified to fit other problems. For instance, applying a signed modulus can extend the network to all real metrics, while increased precision can be achieved by increasing the dimensionality of the latent space, since that allows the network finer control over variations in latent codings. We are truly excited about future applications of our architecture, some of which we will discuss in the next section.

Future work

Based on our current results, this network seems quite promising as a tool for learning metrics over latent representations of data. One simple, yet powerful, modification that could be implemented is the addition of a spatial transform module[12] to the encoder part of the network. For instance, for overlap prediction, the decoder network would decode not the shape itself, but rather a pixelwise transformation field for the first image that would optimally overlap it with the second image. The overlap area itself could then be calculated simply by applying the transformation field. The advantage of this over the current architecture is that it would predict the proper orientation of the two shapes in addition to their maximum area. A similar analogy for drugs would be this network predicting the binding position of the molecules in addition to the binding affinity. This idea could also be aided by a discriminator network learning the autoencoder cost rather than a mathematically defined cost, based on a similar proposition for simpler tasks in Makhzani et al.,[13] which would allow a more robust latent representation, more conducive to decoding a certain property of the molecule.

Another potential development of this network would consist of using it for what is known as drug-autocomplete. Given a latent space generated by our network the continuous metric of binding affinity and the discrete metric of chemical viability, we could theoretically convert the entire space around a certain protein back into data space, yielding tens of potential chemically viable drug candidates with a high binding affinity. Since the number of metrics we can apply is constrained only by the dimensionality of the latent space and computational power, we could use millions of latent dimensions to learn metrics of FDA approval, stability, and even cost.

Acknowledgement

The author would like to thank Gil Alterovitz for his mentorship and guidance, Maksym Korablyov for his mentorship and excellent discussions, the MIT Media Lab for support, and the MIT PRIMES program for this incredible research opportunity.

References

(1) Polishchuk, P. G.; Madzhidov, T. I.; Varnek, A. *Journal of computer-aided molecular design* **2013**, *27*, 675–679.

(2) Kim, S.; Thiessen, P. A.; Bolton, E. E.; Chen, J.; Fu, G.; Gindulyte, A.; Han, L.; He, J.; He, S.; Shoemaker, B. A. e. a. *Nucleic acids research* **2015**, *44*, D1202–D1213.

(3) Kingma, D. P.; Welling, M. *arXiv preprint arXiv:1312.6114* **2013**,

(4) Gómez-Bombarelli, R.; Duvenaud, D.; Hernández-Lobato, J. M.; Aguilera-Iparraguirre, J.; Hirzel, T. D.; Adams, R. P.; Aspuru-Guzik, A. *arXiv preprint arXiv:1610.02415* **2016**,

(5) Kearnes, S.; McCloskey, K.; Berndl, M.; Pande, V.; Riley, P. *Journal of computer-aided molecular design* **2016**, *30*, 595–608.

(6) word2vec authors, word2vec. `https://code.google.com/archive/p/word2vec/`.

(7) Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. *arXiv preprint arXiv:1301.3781* **2013**,

(8) Doersch, C. *arXiv preprint arXiv:1606.05908* **2016**,

(9) Kingma, D.; Ba, J. *arXiv preprint arXiv:1412.6980* **2014**,

(10) AuthorOfThisPaper, A.; Zhao, Q. Affinity Shape Overlap Dataset. `https://electronneutrino.com/affinity/overlap/dataset.zip`.

(11) AuthorOfThisPaper, A. Affinity Shape Matching Dataset. `https://electronneutrino.com/affinity/shapes/datasets/`.

(12) Jaderberg, M.; Simonyan, K.; Zisserman, A. e. a. Spatial transformer networks. Advances in Neural Information Processing Systems. 2015; pp 2017–2025.

(13) Makhzani, A.; Shlens, J.; Jaitly, N.; Goodfellow, I.; Frey, B. *arXiv preprint arXiv:1511.05644* **2015**,

(14) Duvenaud, D. K.; Maclaurin, D.; Iparraguirre, J.; Bombarell, R.; Hirzel, T.; Aspuru-Guzik, A.; Adams, R. P. Convolutional networks on graphs for learning molecular fingerprints. Advances in neural information processing systems. 2015; pp 2224–2232.

(15) Gonczarek, A.; Tomczak, J. M.; Zareba, S.; Kaczmar, J.; Dabrowski, P.; Walczak, M. J. *arXiv preprint arXiv:1610.07187* **2016**,

(16) Pereira, J. C.; Caffarena, E. R.; dos Santos, C. N. *Journal of chemical information and modeling* **2016**, *56*, 2495–2506.

(17) Wan, F.; Zeng, J. *bioRxiv* **2016**, 086033.

(18) Ragoza, M.; Hochuli, J.; Idrobo, E.; Sunseri, J.; Koes, D. R. *arXiv preprint arXiv:1612.02751* **2016**,

(19) Wu, Z.; Ramsundar, B.; Feinberg, E. N.; Gomes, J.; Geniesse, C.; Pappu, A. S.; Leswing, K.; Pande, V. *arXiv preprint arXiv:1703.00564* **2017**,

(20) Altae-Tran, H.; Ramsundar, B.; Pappu, A. S.; Pande, V. *ACS central science* **2017**, *3*, 283–293.

(21) Smith, J. S.; Isayev, O.; Roitberg, A. E. *Chemical Science* **2017**, *8*, 3192–3203.

(22) Faber, F. A.; Hutchison, L.; Huang, B.; Gilmer, J.; Schoenholz, S. S.; Dahl, G. E.; Vinyals, O.; Kearnes, S.; Riley, P. F.; von Lilienfeld, O. A. *arXiv preprint arXiv:1702.05532* **2017**,

(23) Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; Dahl, G. E. *arXiv preprint arXiv:1704.01212* **2017**,

(24) Boscaini, D.; Masci, J.; Rodolà, E.; Bronstein, M. Learning shape correspondence with anisotropic convolutional neural networks. Advances in Neural Information Processing Systems. 2016; pp 3189–3197.

(25) Movshovitz-Attias, Y.; Toshev, A.; Leung, T. K.; Ioffe, S.; Singh, S. *arXiv preprint arXiv:1703.07464* **2017**,

(26) Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. *IEEE transactions on pattern analysis and machine intelligence* **2016**, *38*, 142–158.

(27) Defferrard, M.; Bresson, X.; Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. Advances in Neural Information Processing Systems. 2016; pp 3844–3852.

(28) Qi, C. R.; Su, H.; Mo, K.; Guibas, L. J. *arXiv preprint arXiv:1612.00593* **2016**,

(29) Yi, L.; Su, H.; Guo, X.; Guibas, L. *arXiv preprint arXiv:1612.00606* **2016**,

(30) Graham, B. *arXiv preprint arXiv:1505.02890* **2015**,

(31) Hernández-Lobato, J. M.; Pyzer-Knapp, E.; Aspuru-Guzik, A.; Adams, R. P. Distributed Thompson Sampling for Large-scale Accelerated Exploration of Chemical Space. NIPS Workshop on Bayesian Optimization, Barcelona, Spain. 2016.

(32) Guimaraes, G. L.; Sanchez-Lengeling, B.; Farias, P. L. C.; Aspuru-Guzik, A. *arXiv preprint arXiv:1705.10843* **2017**,

(33) Wallach, I.; Dzamba, M.; Heifets, A. *arXiv preprint arXiv:1510.02855* **2015**,

(34) Xiao, D.; Yang, W.; Beratan, D. N. *The Journal of chemical physics* **2008**, *129*, 044106.

(35) Quiroga, R.; Villarreal, M. A. *PloS one* **2016**, *11*, e0155183.

(36) Arjovsky, M.; Chintala, S.; Bottou, L. *arXiv preprint arXiv:1701.07875* **2017**,

(37) Salimans, T.; Ho, J.; Chen, X.; Sutskever, I. *arXiv preprint arXiv:1703.03864* **2017**,

(38) Bronstein, A. M.; Bronstein, M. M.; Guibas, L. J.; Ovsjanikov, M. *ACM Transactions on Graphics (TOG)* **2011**, *30*, 1.

(39) Rodola, E.; Bronstein, A. M.; Albarelli, A.; Bergamasco, F.; Torsello, A. A game-theoretic approach to deformable shape matching. Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. 2012; pp 182–189.

(40) Wang, F.; Kang, L.; Li, Y. Sketch-based 3d shape retrieval using convolutional neural networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015; pp 1875–1883.

(41) Koch, G.; Zemel, R.; Salakhutdinov, R. Siamese neural networks for one-shot image recognition. ICML Deep Learning Workshop. 2015.

(42) Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D. e. a. Matching networks for one shot learning. Advances in Neural Information Processing Systems. 2016; pp 3630–3638.