



UNIVERSITEIT VAN AMSTERDAM

MSC ARTIFICIAL INTELLIGENCE
MASTER THESIS

Knowledge Generation

by
FLORIAN WOLF
12393339

September 30, 2020

48 Credits
April 2020 - December 2020

Supervisor:
Dr Peter BLOEM
Chiara SPRUIJT

Asessor:
Dr Paul GROTH



Abstract

We generate Knowledge! [1]

1 Introduction

Here comes a beautiful introduction. Promise!

2 Related Work

In this section we will go over related work and relevant background information. The depth of the explanation is adopted to the expected prior knowledge of the reader. The reader is supposed to know the basics of machine learning, including probability theory and basic knowledge on neural networks and their different architectures.

2.1 The Graph VAE – one shot method

VAE

The VAE as first presented by [2] is an unsupervised generative model consisting of an encoder and a decoder. The architecture of the VAE differs from a common autoencoder by having a stochastic module between encoder and decoder. Instead of directly using the output of the encoder, a distribution of the latent space is predicted from which we sample the input to the decoder. The reparameterization trick allows the model to be differentiable. By placing the sampling module outside the model we get a deterministic model which can be backpropagated.

MLP

The Multi-Layer Perceptron (MLP) was one of the first machine learning models. In its basic structure it takes a one dimensional input, fully-connected hidden layer, activation function and finally output layer with normalized predictions. Images or higher dimensional tensors can be processed by flattening them to a one dimensional tensor and adjusting the input dimensions. This makes the MLP a flexible and easy to implement model.(reference)

Graph convolutions

CNNs have shown great results in the field of images classification and object detection. This is due to the fact that a convolution layer takes into account the relation of one pixel to its neighbors. The same holds for graph CNNs where at each convolution the information of each node is passed on as messages to all its neighbors. Each convolution applies an activation function in between the steps. In this case the information is the edges each node has. To process node attributes and edge attributes we have to look at more complex models [3].

RGCN

Relational Graph Convolution Net (RGCN) was presented in [4] for edge prediction. This model takes into account features of nodes. Both the adjacency and the feature matrix are matrix-multiplied with the weight matrix and then with themselves. The resulting vector is a classification of the nodes.

Graph VAE

Now we have all the building blocks for a Graph VAE. The encoder can either be a MLP, a GCNN or an RGCN. The same holds for the decoder with the addition that model architecture needs to be inverted. An version of a Graph VAE presented in [5]. This model combines both the previous methods. The input graph undergoes relational graph convolutions before it is flattened and projected into latent space. After applying the reparameterization trick, a simple MLP decoder is used to regenerate the graph. In addition the model concatenates the input with a target vector y , which represents ????. The same vector is concatenated with the latent tensor. ***Elaborate why they do that***.

Graphs can be generated recursively or in an one-shot approach. This paper uses the second approach and generates the full graph in one go. ***Cite?***

2.2 Permutation Invariance

Permutation invariance refers to the invariance of a permutation of an object. An visual example is the the image generation of numbers. If the loss function of the model would not be permutation invariant, the generated image could show a perfect replica of the input number but due to positional permutation the loss function would penalize the model. OR: An example is in object detection in images. An object can have geometrical permutations such as translation, scale or rotation, none the less the model should be able to detect and classify it. In that case, the model is not limited by permutations and is therefore permutation invariant. In our case the object is a graph and the nodes can take different positions in the adjacency matrix. To detect similarities between graphs we apply graph matching.

Graph matching algorithms

There are various graph matching algorithms. The one we will implement is the max-pooling (Finding Matches in a Haystack: A Max-Pooling Strategy for Graph Matching in the Presence of Outliers).

The max-pooling graph matching algorithm presented by

The algorithm returns a symmetric affinity matrix for all nodes

The resulting similarity matrix gives us X^* which is continuous and therefore useless. To transform it to a discrete X we use the hungarian algorithm (GPU-accelerated Hungarian algorithms for the Linear Assignment Problem)

Hungarian algorithm

The hungarian algorithm is used to find the shortest path within a matrix. This could be the most efficient work-distribution in a cost matrix. Or ...

It consists of four steps of which the last two are repeated until convergence. This algorithm is not scalable. The Munk's algorithm (reference) takes this problem by?? and is scalable.

second option: Compare only graph structure. NX algorithms: Greedy, Shortest path ...

Graph VAE loss The loss is described in [5] as.

In contrast to his implementation we assume, that a node or edge can have none, one or multiple attributes. Therefore our attributes are also not sigmoided and do not sum up to one. This leads to the modification of term $\log F$ and $\log E$ where we do not matrix multiply over the attribute vector but take the BCE as over the rest of the points. KG can have multiple or no attributes vs molecular graphs can be one-hot encoded.

Okay, further we need to treat the $\log_p E$ and $\log_p F$ just like $\log_p A$ and subtract the inverse. Otherwise the model learns to predict very high values only.

A note to the node features, these stay softmaxed and one-hot encoded since we will use them as node labels.

2.3 Knowledge Graphs

Knowledge Graphs are great! The best in the world.

3 Background

4 Methodology

5 Experiments

6 Results

7 Discussion & Future Work

Open Issues

A section where notes will be made on open issues. These issues are meant to be discussed with Peter or Thivyan and ultimately solved. No open issues should remain at the end of November.

7.1 Graph matching

Hungarian algorithm: Should we assume the affinity matrix is a cost or a profit matrix. If we assume it is a cost matrix and $n \neq k$ the algorithm returns an error while padding. If we assume it is a profit matrix and convert it to a profit matrix it can handle any dimension. The shortest path varies with both approaches. Further I assume the shortest path is what we are looking for and mask these entries with 1 and the rest of the matrix with 0.

Diagonal of A is zeros. As soon as I do this, code crashes. Think it is because of divided zero, Martin adds $1e-7$ to the X norm

Do we backpropagate through the graph matching? then all torch functions, otherwise np.

Martin takes 300 iterations for hungarian. also not looping over pairs.

References

1. Kipf, T., van der Pol, E. & Welling, M. Contrastive Learning of Structured World Models. *arXiv:1911.12247 [cs, stat]*. arXiv: 1911.12247. <http://arxiv.org/abs/1911.12247> (2020) (Jan. 2020).

2. Kingma, D. P. & Welling, M. Auto-Encoding Variational Bayes. *arXiv:1312.6114 [cs, stat]*. arXiv: 1312.6114. <http://arxiv.org/abs/1312.6114> (2020) (May 2014).
3. Tiao, L., Elinas, P., Nguyen, H. & Bonilla, E. V. Variational Graph Convolutional Networks. en, 11.
4. Kipf, T. N. & Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. en. *arXiv:1609.02907 [cs, stat]*. arXiv: 1609.02907. <http://arxiv.org/abs/1609.02907> (2020) (Feb. 2017).
5. Simonovsky, M. & Komodakis, N. GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders. *arXiv:1802.03480 [cs]*. arXiv: 1802.03480. <http://arxiv.org/abs/1802.03480> (2020) (Feb. 2018).