AĞLARDADA ÖĞRENME
& PERFORMANSIN ÖTESİ

Bootstrapped Self-Supervised Representation Learning in Graphs

23 Haziran 2021

Etkinlik Zoom üzerinden gerçekleşecektir.
Sunum dili: İngilizce

Shantanyu Thakoor
DeepMind

Michal Valko
DeepMind & INRIA
Why we are after self-supervised learning?

“For AGI we want agents to generalise significantly beyond the specific tasks that they were trained on. ”

Reality check = very limited supervision
... but supervised learning is what ML is good at!

Mastering SSL we equip agents with stronger generalization capabilities.
Agenda for June, 22nd 2021

1. SSL
   - (Modern) SSL
   - How BYOL works
   - ResNet as encoder
   - ImageNet as data

2. Graph Nets
   - Graph Nets as the encoders for graph data

3. BGRL
   - Self-Supervised Learning on Graphs
1 Self-Supervised Learning
Computer Vision Goal

Image

Model

- Classification
- Segmentation
- Object detection
- Depth estimation

Dog
How to train the encoder?
Motivation

Labelled, but costly/few data

Unlabelled, **free** data!
Motivation

Image

Encoder

Downstream network

- Classification
- Segmentation
- Object detection
- Depth estimation

Self-supervised
Free unlabeled data

Supervised
Few labeled data

BYOL
BYOL

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshtaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, Michal Valko
Intuition: Two different views (augmentations) of the same picture should be predictive of each other.

A view of a dog is still a dog, i.e. semantic information is invariant to transformations.

---

SimCLR: Chen et al., A simple framework for contrastive learning of visual representations. ICML. 2020
BYOL main intuition

Image

Views

Predict?
BYOL main intuition

Image → Views → Encoder → Predictor → Prediction
BYOL main intuition

Image → Views → Encoder → Predictor

Prediction → Target
BYOL main intuition

Image

Views

Encoder

Predictor

Random Initialization

Prediction

Random Initialization

Target

Regress
BYOL main intuition

Image  Views  Encoder  Predictor

Random Initialization

Prediction

Regress

Target
BYOL main intuition

Image → Views → Encoder → Predictor

Copy → Prediction

Target
BYOL main intuition

Image → Views → Encoder → Predictor → Prediction → Regress → Target
BYOL main intuition

Image → Views → Encoder → Predictor

Prediction

Copy

Target

Regress
BYOL main intuition
BYOL main intuition

Image | Views | Encoder | Predictor

Copy | Prediction | Regress | Target
BYOL Architecture

Online network

Exponential Moving Average

Target network

\[ q_\theta(z_\theta) \]

\[ \frac{\langle q_\theta(z_\theta), z'_\xi \rangle}{\|q_\theta(z_\theta)\|_2 \cdot \|z'_\xi\|_2} \]

\[ z'_\xi \]
BYOL’s highlights

**Key ingredients:**
- Image transformations.
- Target network.
- Additional predictor on top of online network.

**Interest of the method:**
- Simple training procedure.
- No negative examples [details 3 slides later].
- Work at the embedding level, e.g. no-pseudo labels.
Wait, there has been life before BYOL!
Self-supervised learning

https://paperswithcode.com/sota/self-supervised-image-classification-on

- Generative vs. Predictive
- Contrastive (Positives / Negatives) – [next slide]
  - Positives “corrupted” … otherwise it’s too easy
  - Negatives to rescue
Self-Supervised Learning / Contrastive Losses

Data
\[ \{x\}_i \quad \text{task}_\text{spec} \]

Model
\[ y \approx f_\theta(x) \]

Loss
\[ \mathcal{L}(\theta) = \sum_{i=1}^{N} \log \frac{\exp(f_\theta(\text{aug}(x_i))^T f_\theta(\text{aug}(x_i)))}{\sum_{x'} \exp(f_\theta(\text{aug}(x_i))^T f_\theta(\text{aug}(x')))} \]

Optimisation
\[ \theta^* = \arg \max_\theta \mathcal{L}(\theta) \]
BYOL → Negatives gone!

CONCEPTUAL
- No need to define what is “not an object”
  - for some domains difficult
  - default option may be wrong

SCALABILITY
- for “not an object” we need large batches
- for some domains (graphs..) can be quadratic in sample size

ROBUSTNESS [result in the next slides]
- to augmentation
- to batch size

PS: Prior to BYOL, negatives absent in DeepCluster.
3 Performance of BYOL
Linear Evaluation Protocol on ImageNet

Step 1: Train a “representation” on ImageNet without any labels.

Step 2: On top of the frozen representation, train a linear classifier on ImageNet with label information.
Linear Evaluation Performance on ImageNet

Note: these supervised baselines are from SimCLR (Chen et al., ICML 2020)
Linear Evaluation Performance on ImageNet

Note: these supervised baselines are from SimCLR (Chen & Hinton, ICML 2020)

CPCv2: van den Oord et al., Representation learning with contrastive predictive coding, 2018
AMDIM: Bachman et al., Learning representations by maximizing mutual information across views, 2019
CMC: Tian et al., Contrastive multiview coding, 2019
MoCo: He et al., Momentum contrast for unsupervised visual representation learning, 2019
InfoMin: Tian et al., What makes for good views for contrastive learning, 2020
MoCov2: Jain et al., Improved baselines with momentum contrastive learning, 2020
SimCLR: Chen et al., A simple framework for contrastive learning of visual representations, 2020
Linear Evaluation Performance on ImageNet

**Note:** these supervised baselines are from SimCLR (Chen & Hinton, ICML 2020)

- CPCv2: van den Oord et al., *Representation learning with contrastive predictive coding*. 2018
- AMDIM: Bachman et al., *Learning representations by maximizing mutual information across views*. 2019
- CMC: Tian et al., *Contrastive multiview coding*. 2019
- MoCo: He et al., *Momentum contrast for unsupervised visual representation learning*. 2019
- InfoMin: Tian et al., *What makes for good views for contrastive learning*. 2020
- MoCov2: Jain et al., *Improved baselines with momentum contrastive learning*. 2020
- SimCLR: Chen et al., *A simple framework for contrastive learning of visual representations*. 2020
Further comparison with SimCLR

BYOL outperforms other self-supervised learning methods on the following benchmarks:

- Semi-supervised learning on ImageNet
- Fine-tuning on small classification datasets (such as CIFAR or Flowers)
- Transfer tasks when pretraining on Places365 instead of ImageNet

Summary: BYOL vs. Contrastive methods:

- BYOL is less sensitive to the choice of image transformations
- BYOL is more robust to smaller batch sizes

The code and checkpoints are available: https://github.com/deepmind/deepmind-research
Sensitivity to augmentation choice

BYOL is **predictive** rather than **contrastive** ⇒ lower sensitivity to transformation set.
Self-Supervised Learning on Graphs

Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Mehdi Azabou, Eva L. Dyer, Remi Munos, Petar Veličković, Michal Valko

Thanks to Petar Veličković for help with slides!
Graphs are Everywhere!

- Data with special structure:
  - Nodes = entities
  - Edges = connections between nodes
  - Graphs = collection of nodes with edges
Traffic maps are graphs!

- Transportation networks (e.g. *Google Maps*) naturally modelled as graphs
  - Nodes as intersections, edges as roads
  - Many natural node/edge-level features in this data!
  - Possible task of interest: ETA prediction
Molecules are graphs!

- A very natural way to represent molecules
  - **Atoms** as nodes, **bonds** as edges
  - Features such as **atom type, charge, bond type**...
  - Possible task – predict whether molecule inhibits diseases
How to learn from graphs?

Inputs
$(X, A)$
Graph Neural Networks!
Node-level representations

Inputs $(X, A)$

Latents $(H, A)$

GNN

$z_i = f(h_i)$
Graph-level representations

Inputs $(X, A)$

Latents $(H, A)$

Node classification $z_i = f(h_i)$

Graph classification $z_G = f(\bigoplus_{i \in V} h_i)$
Edge-level representations

Inputs \((X, A)\)

Latents \((H, A)\)

Node classification \(Z_i = f(h_i)\)

Graph classification \(Z_G = f(\bigoplus_{i \in V} h_i)\)

Link prediction \(Z_{ij} = f(h_i, h_j, e_{ij})\)
Graph Neural Networks

- What do we want in a neural network acting over graphs?

- Desiderata:
  - Use graph structure – node/edge features, connections between nodes
  - Not sensitive to order in which node / neighbors are processed – *permutation (equi/in)variant*

- Starting point: let’s take inspiration from image domain!
Convolutional Neural Networks
### Convolutional Neural Networks

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>1</th>
<th>1</th>
<th>1</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

\[
I \quad K \quad I \ast K
\]

The convolution operation is illustrated by sliding a kernel `K` over the input `I` and performing element-wise multiplication and summation.
### Convolutional Neural Networks

$$I \ast K = \begin{bmatrix}
0 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} \ast \begin{bmatrix}
1 & 0 & 1 \\
0 & 1 & 0 \\
1 & 0 & 1
\end{bmatrix} = \begin{bmatrix}
1 & 4 & 3 & 4 & 1 \\
1 & 2 & 4 & 3 & 3 \\
1 & 2 & 3 & 4 & 1 \\
1 & 3 & 3 & 1 & 1 \\
3 & 3 & 1 & 1 & 0
\end{bmatrix}$$
Convolutional Neural Networks

\[ I \times K \rightarrow I \ast K \]
Convolutional Neural Networks

- **Translational invariance**

- Patterns are interesting irrespective of location in image

- **Locality**: neighbouring pixels affect more than distant

- Images are essentially graphs
  - Pixels = nodes arranged in grid connectivity pattern
  - What about **arbitrary** graphs?
Graph Convolutional Networks (GCNs)

- Features of neighbours aggregated with fixed weights, $c_{ij}$
  \[ h_i = \phi \left( x_i, \bigoplus_{j \in N_i} c_{ij} \psi(x_j) \right) \]

- Usually, the weights depend directly on adjacency matrix
  - ChebyNet (Defferrard et al., NeurIPS’16)
  - GCN (Kipf & Welling, ICLR’17)
  - SGC (Wu et al., ICML’19)

- Useful for **homophilous** graphs and **scaling up**
  - When edges encode *label similarity*
The three “flavours” of GNN layers

Convolutional:
\[ h_i = \phi \left( x_i, \bigoplus_{j \in \mathcal{N}_i} c_{ij} \psi(x_j) \right) \]

Attentional:
\[ h_i = \phi \left( x_i, \bigoplus_{j \in \mathcal{N}_i} a(x_i, x_j) \psi(x_j) \right) \]

Message-passing:
\[ h_i = \phi \left( x_i, \bigoplus_{j \in \mathcal{N}_i} \psi(x_i, x_j) \right) \]
Graph Representation Learning

- Goal: Learn meaningful node representations \textit{without supervision}

- Why?
  - Unlabeled data cheaper
  - Pre-training for downstream tasks
  - Auxiliary signal for semi-supervised training
Early methods: Random-walk objectives

- What makes an embedding “good”?
  - Graphs carry interesting **structure**!
  - Good node representations should **preserve** it.

- Simplest notion of graph structure is an **edge**.
  - Features of nodes $i$ and $j$ should be predictive of existence of edge $(i, j)$!
  - Generalize slightly: nodes $i$ and $j$ co-occur in a short random walk
  - Very similar to NLP methods such as *word2vec*

- Dominated unsupervised graph representation learning prior to GNNs!
  - DeepWalk, node2vec
  - Do not scale to large graphs easily, do not work with GNN encoders
Current hot methods: Contrastive

- Contrastive methods
  - Push together similar objects (positive examples)
  - Pull apart dissimilar objects (negative examples)

- Aim: stop contrasting dissimilar objects!

... but why?
Drawbacks of Contrastive Methods

- Case Study #1: Deep Graph Infomax (DGI)
  - Contrast against “negative” graph

Problem #1: Hard to define negative examples

- Many datasets = single graph, no “other” graph
Drawbacks of Contrastive Methods

- Case Study #2: GRACE
  - Positive example = same node across views
  - Negative example = every other pair

Problem #2: All-vs-all contrastive scales quadratically

- Subsampling uniformly is bad
- Choosing smartly is hard
B ootstrapped GR aph L atents (BGRL)
Bootstrapped Graph Latents (BGRL)

- Bootstrap embeddings from each node = no negative examples
Bootstrapped Graph Latents (BGRL)

- Bootstrap embeddings from each node = no negative examples
  - Given a graph

![Graph Diagram]
Bootstrapped Graph Latents (BGRL)

- Bootstrap embeddings from each node = no negative examples
  - Generate 2 augmented views
  - Augmentations = transformations embeddings invariant to
Bootstrapped Graph Latents (BGRL)

- Bootstrap embeddings from each node = no negative examples
  - Two encoders: $\theta_{\text{online}}$, $\Phi_{\text{target}}$
  - Compute $h_1$, $h_2$ respectively
Bootstrapped Graph Latents (BGRL)

- Bootstrap embeddings from each node = no negative examples
  - $h_1$ trained to be *predictive* of $h_2$
  - $p_\theta(h_1) = z_1$
Bootstrapped Graph Latents (BGRL)

- Bootstrap embeddings from each node = no negative examples
  - $z_1$ pushed towards $h_2$
Bootstrapped Graph Latents (BGRL)

- Bootstrap embeddings from each node = no negative examples
  - Flow gradients through $\theta$
Bootstrapped Graph Latents (BGRL)

- Bootstrap embeddings from each node = no negative examples
  - Block gradients through $\Phi$
Bootstrapped Graph Latents (BGRL)

- Bootstrap embeddings from each node = no negative examples
  - $\Phi$ updated as EMA of $\theta$
Bootstrapped Graph Latents (BGRL)

- Adaptation from BYOL – no projector network

- Undesirable/trivial solutions exist (e.g. $\theta = \Phi$)
  - Not obtained as $(\theta, \Phi)$ update does not minimize any loss
Graph Augmentations

- Design decision, perturbations that do not change semantics
- For images, intuitive to design
  - Flipping/cropping/color distortions typically not change class
- For graphs, very unintuitive!
  - Perturb *whole graph*
  - But learn embeddings for *nodes*
  - It would be like augmenting an image but learning pixel-level!
- So simple, cheap augmentations done:
  - Randomly drop certain edges
  - Random node feature masking
  - Not perfect, still open area of research!
Experiments
Experimental Setup

- Node classification, GRACE current best
- Linear evaluation protocol
- Encoders: Graph Convolutional Networks (GCNs)
  BGRL predictor: MLP
- Simple augmentations, masking with fixed probability
Experimental Setup

- Report results relative to randomly initialized GCN

- Very strong baseline!
  - Random GCNs = good inductive bias
  - Linear classifier on top works as normal
  - Surpasses pure supervised in some cases!
Datasets

- **Transductive tasks:**
  - Single graph, all nodes known during training, labels only available for training nodes
  - WikiCS, Coauthor CS/Physics, ogbn-arXiv: citations networks, classify paper topic
  - Amazon Computers/Photos: co-purchase graphs, classify product type

- **Inductive tasks:**
  - Dataset of many graphs, train on some/test on others
  - PPI: dataset of protein–protein interactions, predict biological properties
Experimental Results

- Citations/Co-purchase graphs, $O(10k)$ nodes $\rightarrow$ quadratic possible

- Not only is BGRL $\geq$ other methods, memory usage is 5–10x smaller
Scaling Up to Larger Graphs

- OGB arXiv dataset, 170k nodes
- Subsample $k$ negatives per node for GRACE
  - $k=2 \approx$ BGRL in asymptotic memory

Accuracy Relative to Random Embeddings

![Accuracy Graph]

- DGI
- GRACE($k=2$)
- $k=8$
- $k=32$
- $k=2048$
- BGRL
- Supervised

Equivalent Memory
Pushing Performance on PPI

- PPI: biological networks of protein interactions, $O(50k)$ nodes
  - Huge gap between self-supervised and fully supervised
  - Graph Attentional encoders
Unlocking performance on 1000x larger dataset

- KDD Cup 2021: OGB-LSC challenge, dataset with 240M nodes / 1B edges

- BGRL was key to DeepMind team awarded as Top–3

- BGRL works even with:
  - 1000x larger data
  - Expressive MPNNs
  - Mixing with supervised signals
Conclusions

Main takeaways:
○ BGRL competitive with contrastive methods without negative examples
○ Huge wins in memory and performance in some cases
○ Likely to be more easily applied to larger graphs without design choices

Future directions:
○ Naturally extends to learning graph-level embeddings
○ Experimenting with stronger encoder architectures
○ Research into stronger graph-based augmentations
Thank You!

... Questions?