Vision AI

2022 Arxiv Trends

2022-01

hutom

Content

no.	Paper Title	Correspondence	h-index
1	Robust Contrastive Learning Using Negative Samples with Diminished Semantics	David Jacobs	59
2	Decoupling Zero-shot Semantic Segmentation	Dengxin Dai	31
3	GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models	OpenAI	
4	MGAE: Masked Autoencoders for Self-Supervised Learning on Graphs	Jing Liao	17

GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models

Alex Nichol * Prafulla Dhariwal * Aditya Ramesh * Pranav Shyam Pamela Mishkin Bob McGrew
Ilya Sutskever Mark Chen

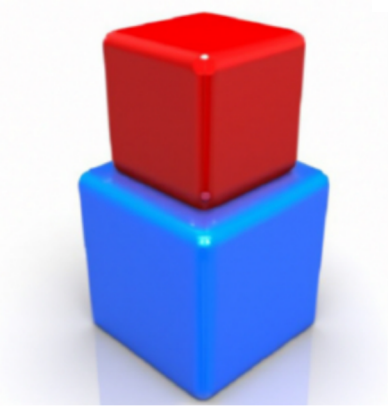
"a pixel art corgi pizza"

Diffusion Model
- text-conditional image synthesis

Two different guidance strategies
1. CLIP guidance
2. classifier-free guidance
   a. human evaluators : photorealism 과 cation similarity 에 더 선호됨

Model fine-tuning
- to perform image inpainting
- enabling powerful text-drivin image editing

github : https://github.com/openai/glide-text2im
- train a smaller model on a filtered dataset
- release the code and weights



"a red cube on top of a blue cube"

"a corgi wearing a red bowtie and a purple party hat"

- Diffusion model : [link]

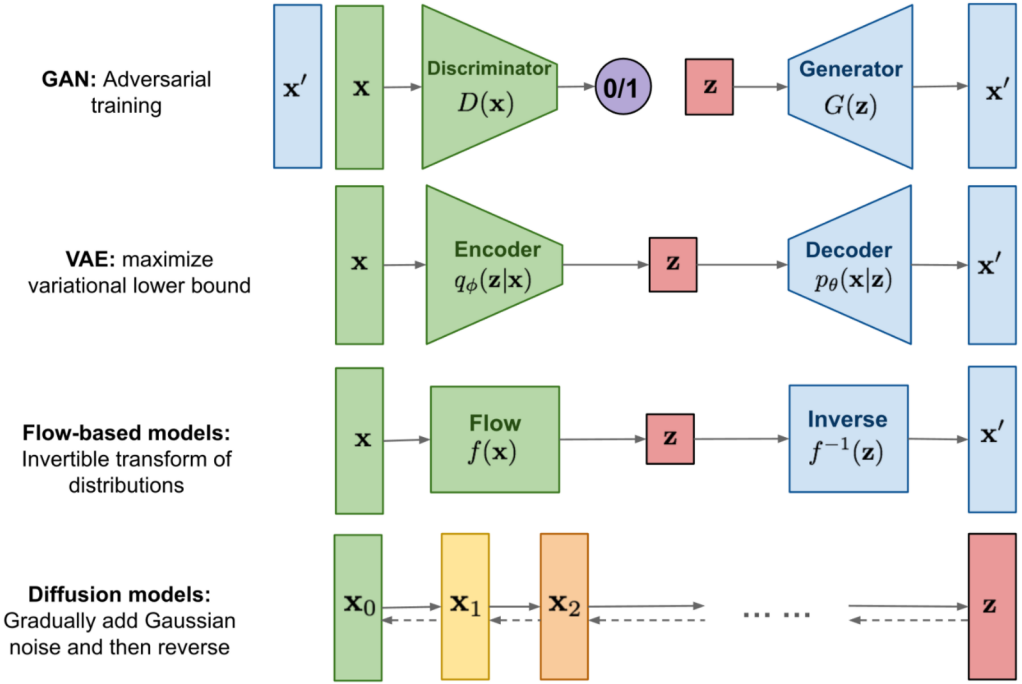


Fig. 1. Overview of different types of generative models.

- Diffusion model : link
- Diffusion process

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \quad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$$

The data sample $\mathbf{x}_0$ gradually loses its distinguishable features as the step t becomes larger. Eventually when $T \to \infty$, $\mathbf{x}_T$ is equivalent to an isotropic Gaussian distribution.

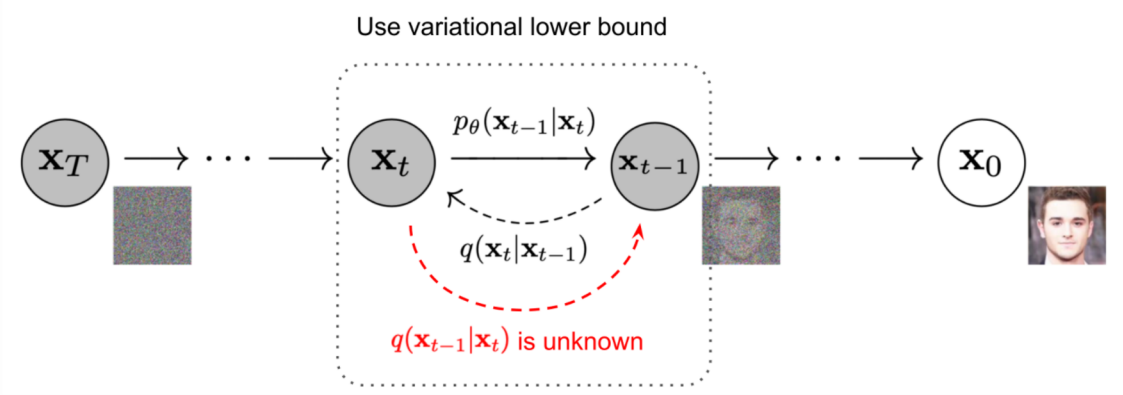


Fig. 2. The Markov chain of forward (reverse) diffusion process of generating a sample by slowly adding (removing) noise. (Image source: Ho et al. 2020 with a few additional annotations)

- Diffusion model : link
- Reverse process

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \quad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$

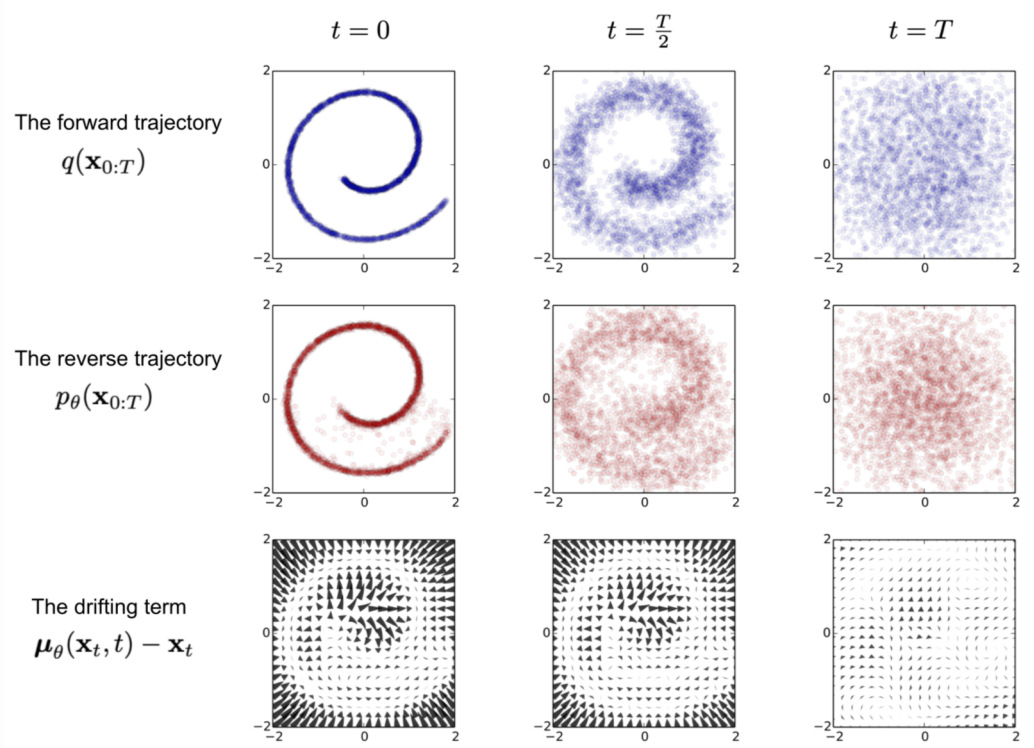


Fig. 3. An example of training a diffusion model for modeling a 2D swiss roll data.
(Image source: Sohl-Dickstein et al., 2015)

The components of the final models we will evaluate

- diffusion
- classifier-free guidance
- CLIP guidance.

2.1. Diffusion Models

We consider the Gaussian diffusion models introduced by Sohl-Dickstein et al. (2015) and improved by Song & Ermon (2020b); Ho et al. (2020). Given a sample from the data distribution $x_0 \sim q(x_0)$, we produce a Markov chain of latent variables $x_1, ..., x_T$ by progressively adding Gaussian noise to the sample:

$$q(x_t|x_{t-1}) \coloneqq \mathcal{N}(x_t; \sqrt{\alpha_t}x_{t-1}, (1-\alpha_t)\mathcal{I})$$

2.2. Guided Diffusion

Dhariwal & Nichol (2021) find that samples from class-conditional diffusion models can often be improved with classifier guidance, where a class-conditional diffusion model with mean $\mu_\theta(x_t|y)$ and variance $\Sigma_\theta(x_t|y)$ is additively perturbed by the gradient of the log-probability $\log p_\phi(y|x_t)$ of a target class y predicted by a classifier. The resulting new perturbed mean $\hat{\mu}_\theta(x_t|y)$ is given by

$$\hat{\mu}_\theta(x_t|y) = \mu_\theta(x_t|y) + s \cdot \Sigma_\theta(x_t|y)\nabla_{x_t} \log p_\phi(y|x_t)$$

2.3. Classifier-free guidance

Ho & Salimans (2021) recently proposed classifier-free guidance, a technique for guiding diffusion models that does not require a separate classifier model to be trained. For classifier-free guidance, the label y in a class-conditional diffusion model $\epsilon_\theta(x_t|y)$ is replaced with a null label $\emptyset$ with a fixed probability during training. During sampling, the output of the model is extrapolated further in the direction of $\epsilon_\theta(x_t|y)$ and away from $\epsilon_\theta(x_t|\emptyset)$ as follows:

$$\hat{\epsilon}_\theta(x_t|y) = \epsilon_\theta(x_t|\emptyset) + s \cdot (\epsilon_\theta(x_t|y) - \epsilon_\theta(x_t|\emptyset))$$

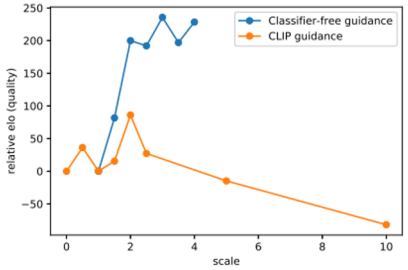
Here $s \geq 1$ is the guidance scale. This functional form is inspired by the implicit classifier

$$p^i(y|x_t) \propto \frac{p(x_t|y)}{p(x_t)}$$

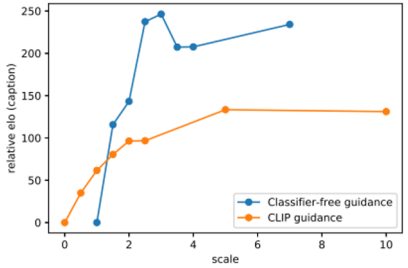
whose gradient can be written in terms of the true scores ϵ^*

$$\begin{aligned}\nabla_{x_t} \log p^i(x_t|y) &\propto \nabla_{x_t} \log p(x_t|y) - \nabla_{x_t} \log p(x_t) \\ &\propto \epsilon^*(x_t|y) - \epsilon^*(x_t)\end{aligned}$$

- Classifier-free guidance 의 성능(blue)이 CLIP Guidance(orange) 보다 더 좋음.



(a) Photorealism



(b) Caption Similarity

Figure 7. Elo scores from human evaluations for finding the optimal guidance scales for classifier-free guidance and CLIP guidance. The classifier-free guidance and CLIP guidance comparisons were performed separately, but can be super-imposed onto the same graph my normalizing for the Elo score of unguided sampling.

Table 1. Elo scores resulting from a human evaluation of unguided diffusion sampling, classifier-free guidance, and CLIP guidance on MS-COCO validation prompts at 256 × 256 resolution. For classifier-free guidance, we use scale 3.0, and for CLIP guidance scale 2.0. See Appendix A.1 for more details on how Elo scores are computed.

Guidance	Photorealism	Caption
Unguided	-88.6	-106.2
CLIP guidance	-73.2	29.3
Classifier-free guidance	**82.7**	**110.9**

- 포토리얼리스틱

"a surrealist dream-like oil painting by salvador dalí of a cat playing checkers"

"a professional photo of a sunset behind the grand canyon"

"a high-quality oil painting of a psychedelic hamster dragon"

"an illustration of albert einstein wearing a superhero costume"

"a crayon drawing of a space elevator"

"a futuristic city in synthwave style"

"a pixel art corgi pizza"

"a fog rolling into new york"

- 꼭 잘 되는 것만은 아니다. failure cases

Figure 8. Failure cases of GLIDE when prompted for certain unusual objects or scenarios.

Decoupling Zero-Shot Semantic Segmentation

Jian Ding[1,2], Nan Xue[1], Gui-Song Xia[1], Dengxin Dai[2]
[1]CAPTAIN, Wuhan University, China
[2]MPI for Informatics, Germany
{jian.ding, xuenan, guisong.xia}@whu.edu.cn, ddai@mpi-inf.mpg.de

- **zero-shot sementic segmentation (ZS3)의 목적: unseen class를 pixel-level 단위로 ?**
  - 기존 pixel-level zero-shot classification problem으로 작동
  - Unseen class는 어떻게 예측?
    ⇒ 기존의 seen class의 지식을 unseen class로 예측하는 지식으로
    pretrained 된 language model 을 사용했다고 함.
  - 근데 vision-language model이 엄청 큰 포텐셜이지만 기존 구조는 이식하기 힘듬

⇒ 이에 ZS3 를 decoploing, 2가지 subtask를 거침

1. a class-agnostic grouping task to group the pixels into semgnets
2. a zero-shot classification task on segmetns

- **제안한 formulation은 ZegFormer**
- COCO-stuff 나 PASCAL VOC에서 outperform

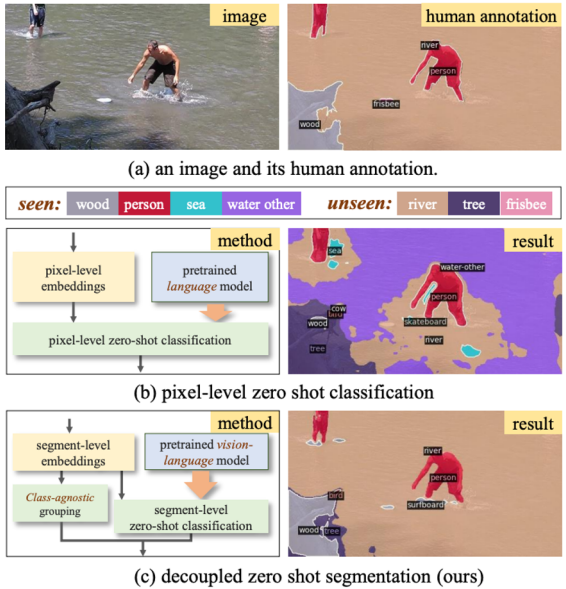


Figure 1. ZS3 aims to train a model merely on *seen classes* and generalize it to classes that have not been seen in the training (*unseen classes*). Existing methods formulate it as a pixel-level zero-shot classification problem **(b)**, and use semantic features from a *language* model to transfer the knowledge from seen classes to unseen ones. In contrast, as in **(c)**, we decouple ZS3 into two sub-tasks: 1) A *class-agnostic* grouping and 2) A *segment-level* zero-shot classification, which enables us to take full advantage of the pre-trained *vision-language* model.

Methology

Decoupling Formulation of ZS3

Actually, if we denote S as the category set of the annotated dataset $\mathbf{D}$, *i.e.*, the seen classes, and E as those appearing in the testing process, we have three types of settings for semantic segmentation,

- fully-supervised semantic segmentation: $E \subseteq S$,
- zero-shot semantic segmentation (ZS3): $S \cap E = \emptyset$,
- generalized ZS3 (GZS3) problem: $S \subset E$.

In this paper, we mainly address the problem with the GZS3 setting, and denote $U = E - E \cap S$ as the set of unseen classes.

{S} : seen class (class in train data)

{E}: class in test data

{U}: unseen data class

⇒ GZS3 setting 으로 문제를 풀꺼라 Unseen class 정의를 저렇게 했다.

- methology 방법이라고 한다면,,

1. generate a set of segment-level embeddings
2. project them for class-agnostic grouping
3. segment-level zero-shot classification by two parallel layers

⇒ 이 때, pretrained image encoder 는 segment classification 하는데 사용되고

⇒ two segment-level classification scores 가 최종적으로 fused 되어 results

ZegFormer

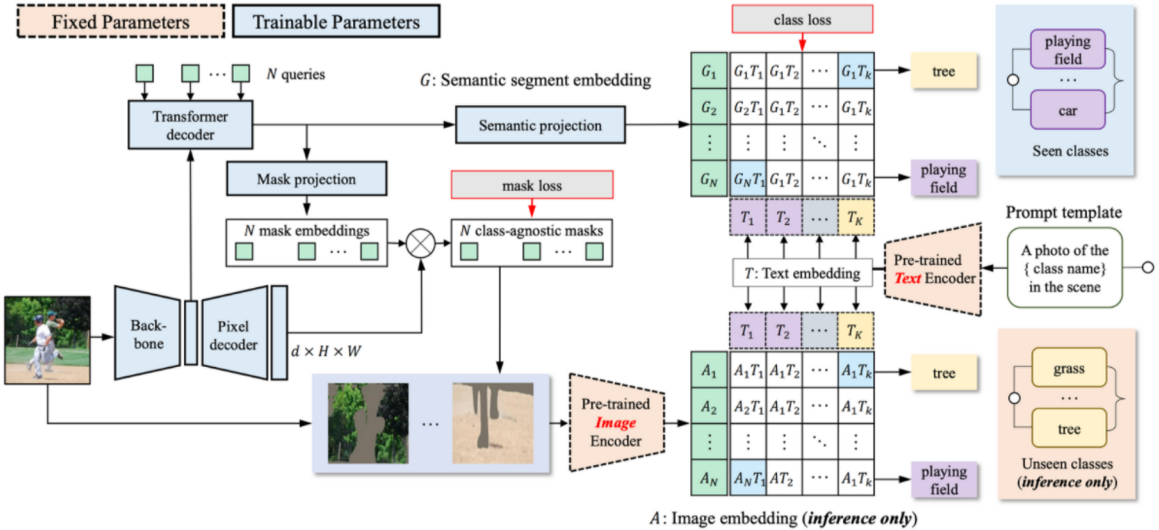


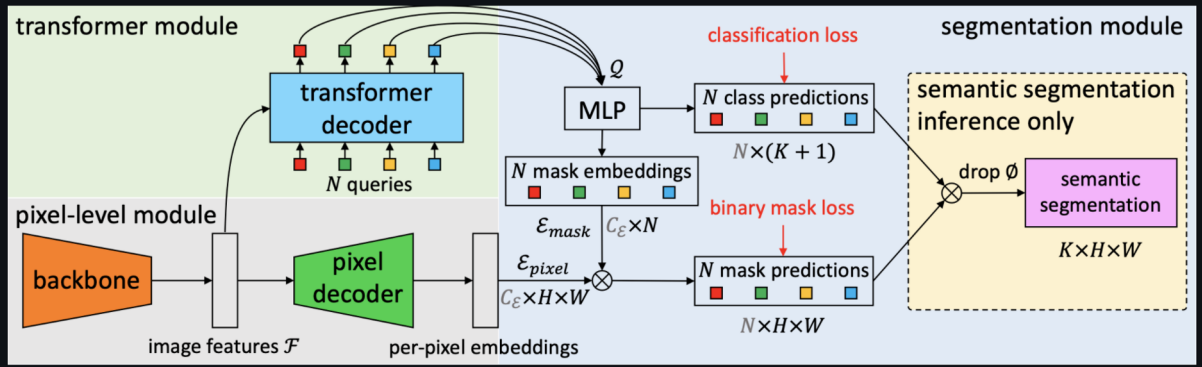
Figure 2. The pipeline of our proposed ZegFormer for zero-shot semantic segmentation. We first feed N queries and feature maps to a transformer decoder to generate N *segment embeddings*. We then feed each segment embedding to a mask projection layer and a semantic projection layer to obtain a mask embedding and a semantic embedding. Mask embedding is multiplied with the output of pixel decoder to generate a *class-agnostic binary mask*, while The semantic embedding is classified by the text embeddings. The text embeddings are generated by putting the class names into a prompt template and then feeding them to a text encoder of a vision-language model. During training, only the seen classes are used to train the segment-level classification head. During inference, both the text embeddings of *seen* and *unseen* classes are used for segment-level classification. We can obtain two segment-level classification scores with semantic segment embeddings and image embeddings. Finally, we fuse the these two classification scores as our final class prediction of segments.

- MaskFormer ?

MaskFormer: Per-Pixel Classification is Not All You Need for Semantic Segmentation

Bowen Cheng, Alexander G. Schwing, Alexander Kirillov

[arXiv] [Project] [BibTeX]



Mask2Former

Checkout Mask2Former, a universal architecture based on MaskFormer meta-architecture that achieves SOTA on panoptic, instance and semantic segmentation across four popular datasets (ADE20K, Cityscapes, COCO, Mapillary Vistas).

Table 4. Comparison with the previous GZS3 methods on PASCAL VOC and COCO-Stuff. The "Seen", "Unseen", and "Harmonic" denote mIoU of seen classes, unseen classes, and their harmonic mean. The STRICT [42] proposed a self-training strategy and applied it to SPNet [54]. The numbers of STRICT and SPNet (w/ self-training) are from [42]. Other numbers are from their original papers.

				PASCAL VOC			COCO-Stuff		
Methods	Type	w/ Self-train.	Re-train. for new classes?	Seen	Unseen	Harmonic	Seen	Unseen	Harmonic
SPNet [54]	discriminative	✗	✗	78.0	15.6	26.1	35.2	8.7	14.0
ZS3 [7]	generative	✗	✓	77.3	17.7	28.7	34.7	9.5	15.0
CaGNet [20]	generative	✗	✓	78.4	26.6	39.7	33.5	12.2	18.2
SIGN [13]	generative	✗	✓	75.4	28.9	41.7	32.3	15.5	20.9
Joint [5]	discriminative	✗	✗	77.7	32.5	45.9	-	-	-
ZS3 [7]	generative	✓	✓	78.0	21.2	33.3	34.9	10.6	16.2
CaGNet [20]	generative	✓	✓	78.6	30.3	43.7	35.6	13.4	19.5
SIGN [13]	generative	✓	✓	83.5	41.3	55.3	31.9	17.5	22.6
SPNet [42]	discriminative	✓	✓	77.8	25.8	38.8	34.6	26.9	30.3
STRICT [42]	discriminative	✓	✓	82.7	35.6	49.8	35.3	30.3	32.6
ZegFormer	discriminative	✗	✗	**81.3**	**76.8**	**79.0**	**36.6**	**33.2**	**34.8**

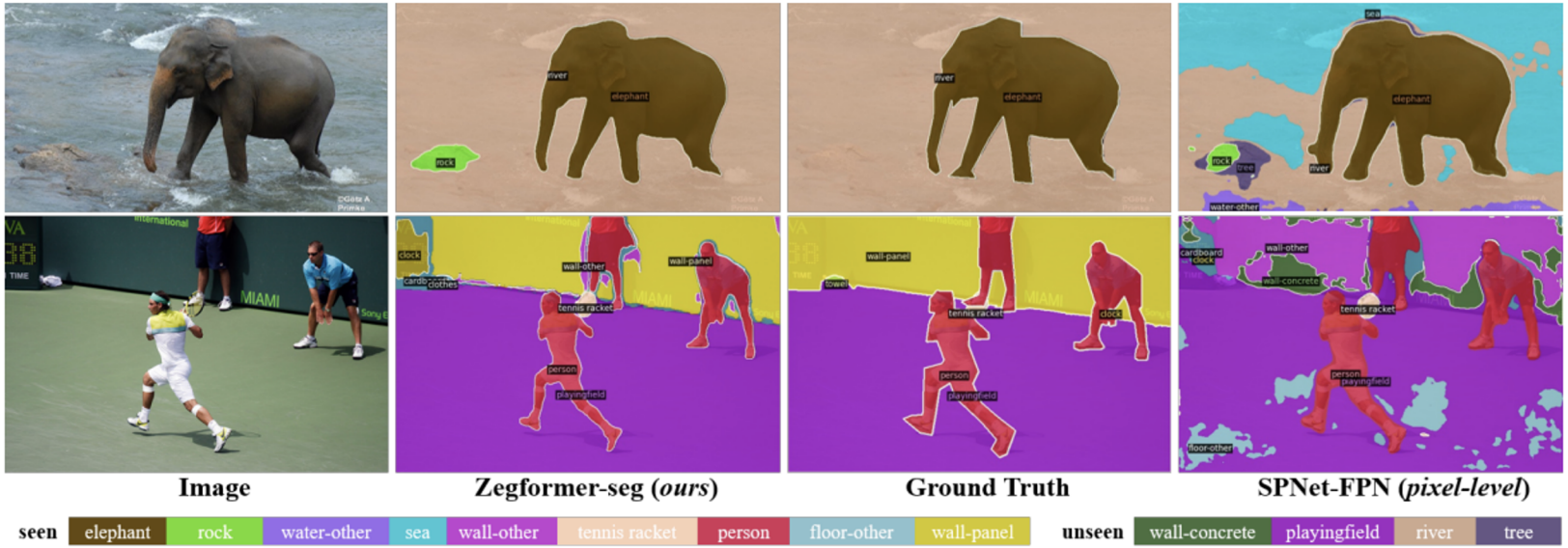


Figure 5. Results on COCO-Stuff, using 171 class names in COCO-Stuff to generate text embeddings. ZegFormer-seg (our decoupling formulation of ZS3) is better than SPNet-FPN (pixel-level zero-shot classification) in segmenting unseen categories.

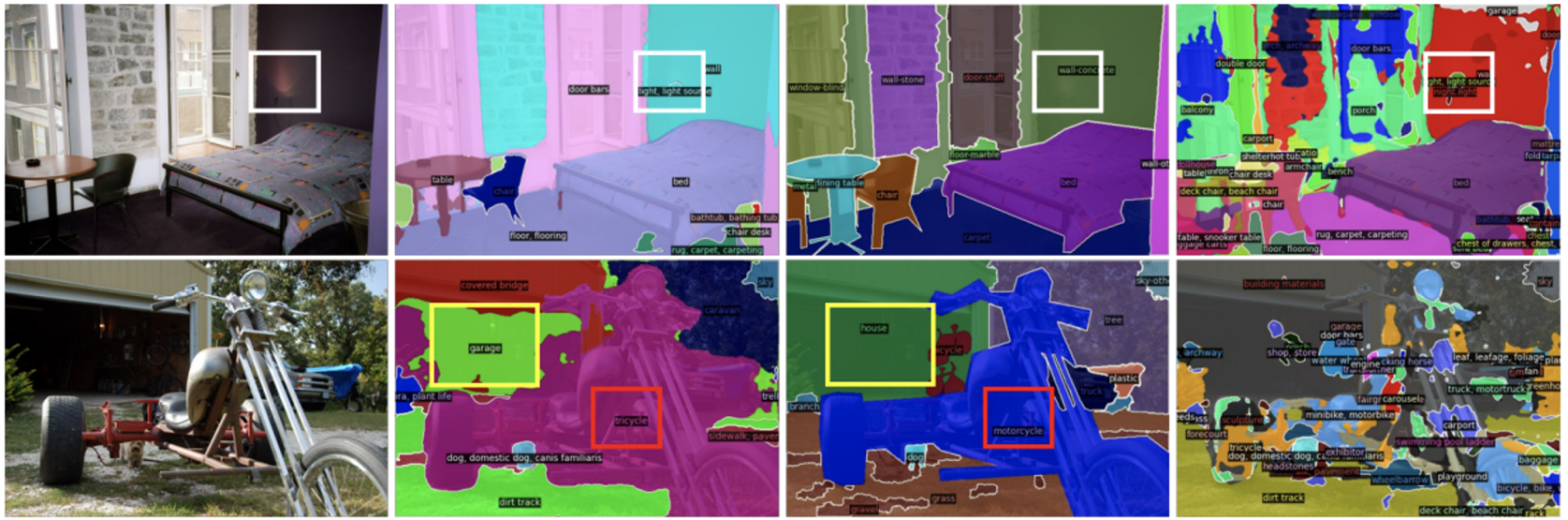
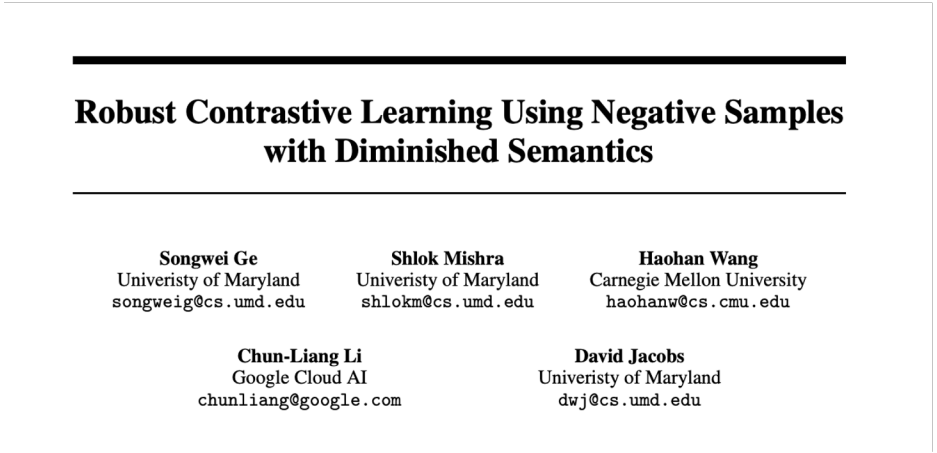


Figure 6. Results on COCO-Stuff, using 847 class names from ADE20k-Full to generate text embeddings. We can see that the SPNet-FPN (*pixel-level zero-shot classification baseline*) is very unstable when the number of unseen classes is large. We can also see that the set of 847 class names provide richer information than the set of 171 class names in COCO-Stuff. For example, in the yellow box of the second row, the predicted "garage" for a segment is labeled as "house" in COCO-Stuff. We provide more visualization results in our appendix.



Robust Contrastive Learning Using Negative Samples with Diminished Semantics

Songwei Ge
Univeristy of Maryland
songweig@cs.umd.edu

Shlok Mishra
Univeristy of Maryland
shlokm@cs.umd.edu

Haohan Wang
Carnegie Mellon University
haohanw@cs.cmu.edu

Chun-Liang Li
Google Cloud AI
chunliang@google.com

David Jacobs
Univeristy of Maryland
dwj@cs.umd.edu

- Unsupervised learning의 발전은 effective Contrastive learning methods
- 반면, CNN은 사람이 인지하지 못하는 low-level feature를 의존하는 경향 (학습하는 경향?)
- 이러한 의존성은 model의 robustness의 부재를 유도
  - Image perturbations
  - domain shift
- 해당 논문, 이러한 의존성 정보를 덜 학습하면서 robust representation 할 수 있도록 함
  - 어떻게? negative sample 을 신중하게 생성함으로서.
- 사실 Contrastive learning에서 positive pair를 utilize하는 것은 학습 중 superfical(표면) feature들을 perturbing하면서 조작하게 됨.
  - 마찬가지로, 우리는 sementic feature대신 불필요한 특징만 보존하는 역방향으로 negative sample을 생성할 것을 제안

Introduction

Figure 1: We propose to construct negative samples (NS) from input images for contrastive learning with augmentations that only preserve non-semantic information such as texture and local features.

1. texture-based patch based aumentation 으로 negative sample 하는 것은 contrastive learning 의 generalization을 향상
2. ImageNet-Texture dataset 제공 (texture version of ImageNet images generated by texture synthesis tools)
3. fine-grained anaylsis on the shape-texture trade-off of CNNs

with Diminished Semantics

1. NCE loss extend version

- z_i, z_p: representation of the query sample x_i and positive sample x_p generated from the same input image with augmentations that preserve semantics.
- z_n: representation of the standard negative sample
  - MoCo(extracted from memory bank) and SimCLR(other images in the current batch)
- z_ns: representation of the proposed negative samples x_ns (contains particluar non-sementic feature of input image)

$$\mathcal{L}_{\text{NCE}} = -\sum_{i \in I} \log \frac{\exp\left(\boldsymbol{z}_i^T \boldsymbol{z}_p / \tau\right)}{\exp\left(\boldsymbol{z}_i^T \boldsymbol{z}_p / \tau\right) + \exp\left(\alpha \boldsymbol{z}_i^T \boldsymbol{z}_{ns} / \tau\right) + \sum_{n \in \mathcal{N}} \exp\left(\boldsymbol{z}_i^T \boldsymbol{z}_n / \tau\right)}, \qquad (1)$$

  - a (alpha) 에 scaling parrameter넣어서 non-semantic representation에 강한 패널티주어 조정

2. BYOL version

- 원래 BYOL 의 경우는 standard negative sample 에 의존하지 않고 대신 positive pairs 의 aggrement 를 최대화 시키는데 집중하는 구조 ⇒ 여기서는 z_ns의 관계도 추가
- BYOL 설명 - 학습되지 않은 target model 의 출력(A model의 predictions을 label로하여) labling 하여 online networks 를 더 잘 학습시키기 (해당 predictions으로 B모델을 더 잘 predict 하기!): bootstrapping
- z_i 와 z_p는 target netowrk 와 online network 의 presentation ? (z_i 와 z_ns도 negative sample augmentation만 바뀌고 똑같지 않을까?) ⇒ positive에 대해서는 최대한 동일하게, negative도 동일하게 하는 것 같은데, 앞에 -alpha 니까 멀리하는것같은데??????

Methods like BYOL [16] do not explicitly rely on negative samples. Nevertheless, BYOL adapts the loss to maximize the agreement of positive pairs. Therefore, we explicitly use the non-semantic negative sample with their loss to minimize its similarity to the query sample:

$$\mathcal{L}_{\text{BYOL}} = \|\boldsymbol{z}_i - \boldsymbol{z}_p\| - \alpha\|\boldsymbol{z}_i - \boldsymbol{z}_{ns}\| = 2 - 2\alpha - 2\boldsymbol{z}_i^T\boldsymbol{z}_p + 2\alpha\boldsymbol{z}_i^T\boldsymbol{z}_{ns}. \quad (2)$$

We overload α to be the parameter that controls the penalty on the similarity between the representations of input image and its non-semantic version under BYOL, with similar intention as MoCo and SimCLR above. To minimize either $\mathcal{L}_{\text{NCE}}$ or $\mathcal{L}_{\text{BYOL}}$, the encoder must learn features from $\boldsymbol{x}_i$ that are not contained in $\boldsymbol{x}_{ns}$ but shared with $\boldsymbol{x}_p$.

with Diminished Semantics

3. Texture-baed negative sample generation

- local structure 를 최대한 보존하는 방향으로 텍스터 합성
- 2개 패칭 ⇒ off the shelf texture synthesis 알고리즘 사용
  - one patch 는 center 에서 (object 의 reflct 추출?) - 이미지넷 같은경우는 중앙에 객체가 있는 데이터
  - the other 은 random location 에서 (background의 reflect 추출?)

4. Patch-based

- simple solution은 아닌 것 같아 보인다고 하네.. ? 이전 연구에서 이미지와 패치가 잘린 이미지가 모델에 의해서 잘 구별되지 않는 다는 점
  - 그래서 이런현상을 줄이고자 여러개의 스케일로 patcing ? ⇒ model 이 다양한 스케일에서 texture 를 보게하기 위해서

5. How hard are the texture-based and patch-based negative samples?

- 사실 C-learning 에서 hard negative sample 하는 것이 제일 중요하지 않음?
- 그래서 우리가 제안한 negative sample 이 standard NS 보다 얼마나 hard 한지를 보여주려고 한다.

  ⇒ query sample 과 cosine similiaty 구해봄 (Random sampling, MoCO-v2, 200epoch, ImageNet-1K)

  ⇒ 그러게 제안한 NS들이 hard 해보이네..

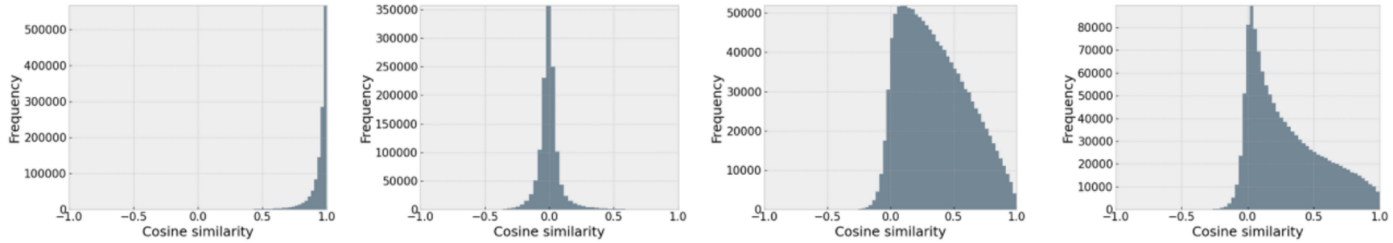


Figure 2: The histogram of cosine similarity between the representations of query sample and its paired samples, namely positive sample and standard, texture-based, and patch-based negative samples (NS), using MoCo-v2 model trained on the ImageNet-1K dataset for 200 epochs.

with Diminished Semantics

Experiments

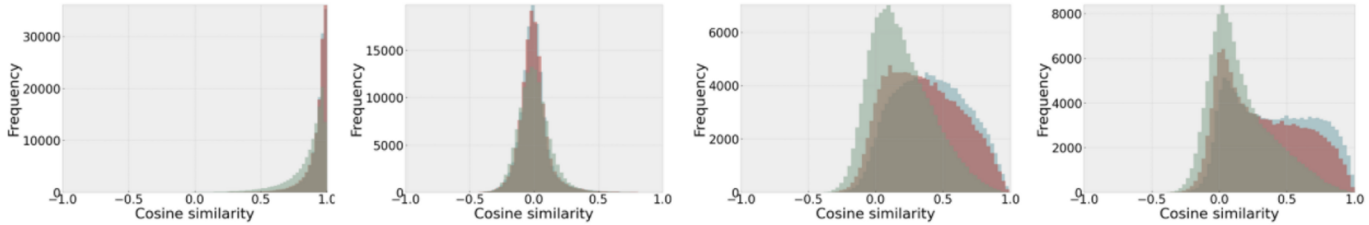
- 2개의 contrasirve methods 가지고 evaluation (non-semanric negative sampling을 가지고)
  - MoCO, BYOL
- 결과는 in-domain과 OOD(out-of-domain) dataset 에서 실험, domain shift 가 잘 일어나는지?
  - texture base
    - in-domain에서도 성능떨어짐, 근데 OOD에서도 성능 개선 (contrastive learning generalizes better under domain shifts)
    - texture 이미지에 포함된 정보는 두가지 고정 patches에 대한 access로 인해 정보가 제한되지 않았나??
  - patch base: 둘다 오름. (요게 좀 크게 오르는 것 같은데?)

with Diminished

3.1 ImageNet-100

	ImageNet	ImageNet-C	ImageNet-S	Stylized-ImageNet	ImageNet-R
MoCo-v2 - $k = 16384$	77.88±0.28	43.08±0.27	28.24±0.58	16.20±0.55	32.92±0.12
+ Texture-based - $\alpha = 2$	77.76±0.17	43.58±0.33	29.11±0.39	16.59±0.17	33.36±0.15
+ Patch-based - $\alpha = 2$	**79.35**±0.12	**45.13**±0.35	31.76±0.88	17.37±0.19	34.78±0.15
+ Patch-based - $\alpha = 3$	75.58±0.52	44.45±0.15	**34.03**±0.58	**18.60**±0.26	**36.89**±0.11
MoCo-v2 - $k = 8192$	77.73±0.38	43.22±0.39	28.45±0.36	16.83±0.12	33.19±0.44
+ Patch-based - $\alpha = 2$	**79.54**±0.32	**45.48**±0.20	**33.36**±0.45	**17.81**±0.32	**36.31**±0.37
MoCo-v2*	80.00±0.14	45.15±0.42	30.38±0.30	16.68±0.39	30.38±0.30
+ IFM [44] - $\epsilon = 0.05$	80.86	47.36	31.35	18.18	36.79
+ IFM [44] - $\epsilon = 0.1$	81.22	47.46	31.87	18.42	37.23
+ IFM [44] - $\epsilon = 0.2$	81.02	47.19	31.55	**18.68**	37.14
+ Patch-based - $\alpha = 2$	**81.49**±0.11	**47.48**±0.20	**34.20**±0.40	17.95±0.41	**38.45**±0.19
BYOL	78.76±0.28	44.43±0.35	35.84±0.38	15.01±0.19	39.53±0.51
+ Patch-based - $\alpha = 0.05$	**78.81**±0.33	**44.60**±0.21	**36.76**±0.51	**15.52**±0.22	**41.16**±0.39
InsDis [53]	68.52	28.93	16.67	9.86	19.60
CMC [48]	79.34	39.28	24.04	13.88	32.68
InfoMin [49]	82.74	48.87	38.43	18.14	40.68
Supervised	86.26	49.17	34.95	21.20	39.76

Table 1: Top-1 accuracy on the ImageNet-100 dataset and its OOD variants. We consider the supervised baseline as well as several self-supervised baselines including MoCo-v2, BYOL, InsDis, CMC, and InfoMIN. For our main comparison using MoCo-v2 and BYOL, we also report the standard deviation of 3 runs. For MoCo models, k represents the size of the memory bank. We use * to denote the experiments that use the training setting in a concurrent work IFM [44].



(a) Positive Sample (b) Standard NS (c) Texture-based NS (d) Patch-based NS

Figure 3: Histogram of cosine similarity between the representations of query sample and its paired positive sample, standard, texture-based, and patch-based negative samples (NS), using models trained without (blue) and with patch-based negative samples (red: $\alpha = 2$, green: $\alpha = 3$).

- patch based negative samples 사용했을 때 안했을 때 consine similiarty 가 어떻게 변했는지 비교해 보자
- figure 3.d 보면 out methods가 input images와 patch-based NS와의 similiarty를 줄임 (alpha 0→ 2→ 3 갈수록 평균 simialirty가 줄어든다고 함)
- figure 3.c 보면 texture-based NS도 similiaty 가 줄어듬. (patch based simility 에 focuse했는데도 불구하고)

◦ ImagaeNet - 1k도 잘되는 듯 함?

with Diminished Semantics

- ImagaeNet - 1k도 잘되는 듯 함?

3.2 ImageNet-1K

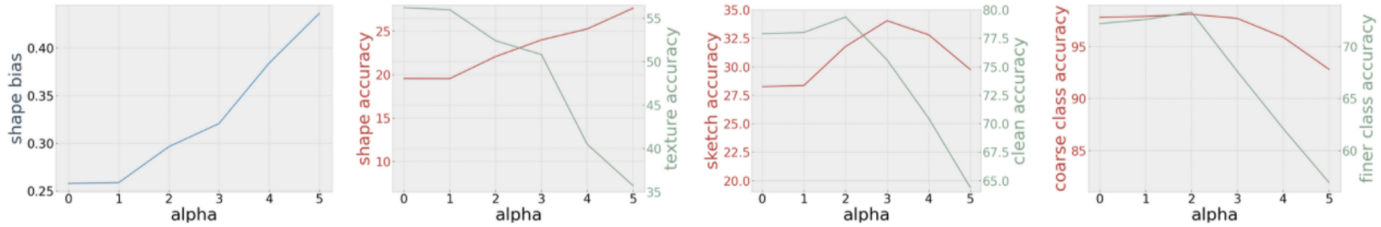
	ImageNet	ImageNet-C	ImageNet-S	Stylized-ImageNet	ImageNet-R
MoCo-v2 [8]	67.60	87.7	17.47	5.55	27.81
+ MoCHi [31]	67.56	88.7	16.32	5.94	25.71
+ Patch-base NS - $\alpha = 2$	**67.92**	**87.6**	**18.58**	**6.34**	**28.95**

Table 2: Top-1 accuracy on the ImageNet-1K dataset and its sketch, stylized, rendition variants, and mCE on the ImageNet-C dataset.

Discussion

Controlling the shape-texture trade-off with alpha

- CNN이 texture feature 에 over-reliant 해보인다.
- Contrasrive learning 에서 non-semantic negatives 를 제공해주는 것만으로 이러한 relizance 를 줄이는 듯 보인다. (약간 trade-off 같음.)



(a) Shape-bias (b) Shape-texture accuracy (c) Sketch-clean accuracy (d) Coarse-finer accuracy

Figure 6: Larger α monotonically increases the model bias to shape features over texture features. Model performance is impacted by such a trade-off differently under different settings. In all scenarios, slightly calibrated shape bias improves model performance. The test settings represented in the red lines gain more from the increased shape bias than the settings represented in the green lines.

Closing Remarks

conclusion

- CNN에서 discriminiative feature 를 배우는 경향이 있음. (domain shift에서 취약)
  - 적절한 negative sample 을 생성하여 대조학습에서의 regluratzation을 했다.
  - patch, texture based 방법
  - 그래서 contrasirve learning 에서의 negatvie samples 의 효과에 대해서 rethink 해봐라

Limitations

- texture feature를 통해 model의 bias를 calibration 할 수 있다는 걸 보여줌
- 하지만 superfical feature 에 대해서만 확인한 것 (like vision)
- 아마 NLP domain에서는 해당 아이디어를 어떻게 녹여내야 할지 모르겠음.

https://arxiv.org/pdf/2201.02534v1.pdf

MGAE: Masked Autoencoders for Self-Supervised Learning on Graphs

Qiaoyu Tan
Texas A&M University
qytan@tamu.edu

Ninghao Liu
University of Georgia
ninghao.liu@uga.edu

Xiao Huang
The Hong Kong Polytechnic University
xiaohuang@comp.polyu.edu.hk

Rui Chen
Samsung Research America
rui.chen1@samsung.com

Soo-Hyun Choi
Samsung Research America
soohyunc@gmail.com

Xia Hu
Rice University
xia.hu@rice.edu

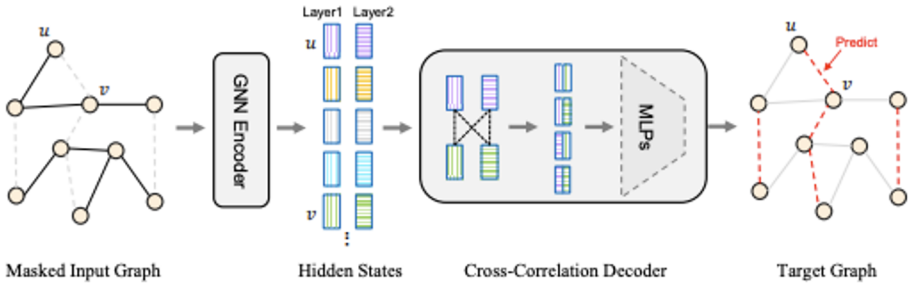
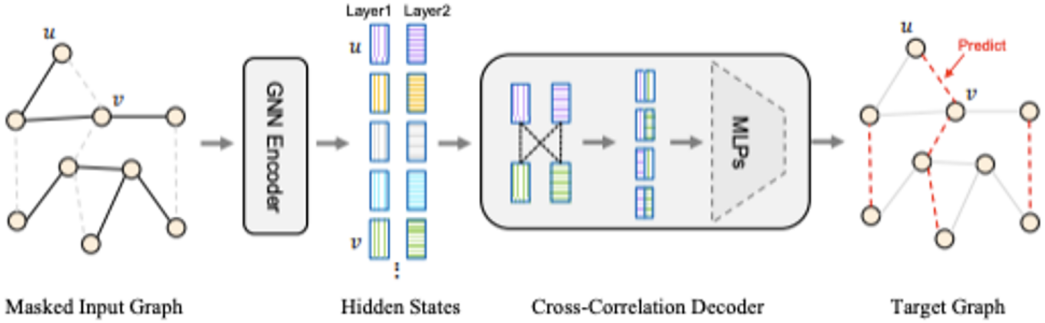


Figure 1: **The proposed MGAE architecture**. Given a graph, a large random subset of edges are masked. The GNN encoder is applied to the remaining edges to produce hidden representations. The representations are then fed into a tailored decoder to reconstruct the masked edges during training, which captures the cross-correlation between end nodes with multi-granularity features.

<Abstract>

- masked graph autoencoder (MGAE) framework to perform effective learning on graph structure data 제시
- large proportion of edges를 random mask하고 training중에 masked된 missing edges를 재구성
- two core designs
  - masking a high ratio of the input graph structure (ex. 70%) yields a nontrivial and meaningful self-supervisory task that benefits downstream applications
  - masked graph에서 message propagation를 수행하기 위해 graph neural network(GNN)을 encoder로, large number of masked edges를 reconstruct하기 위해 cross-correlation decoder 사용
- Planetoid and OGB benchmarks (open dataset) 사용했을때, MGAE가 link prediction and node classification부분에서 generally performs better than state-of-the-art unsupervised learning

https://arxiv.org/pdf/2201.02534v1.pdf



- reconstructs the original network structure given only partially observed edges
- 기존 graph autoencoders, MGAE has a GNN encoder(각 node를 latent embedding으로 mapping) and decoder(latent space에서 input graph의 original link를 recover)
- difference:
  - 1) allow the GNN encoder to operate only on partial network structure (i.e., a portion of edges are masked)
  - 2) cross-correlation decoder는 multigranularity(다중 격자성)에서 anchor edge node의 head와 tail의 cross-correlation(교차 상관 관계..?)를 capture할 수 있다

- **network masking, GNN encoder, cross-correlation decoder, and the reconstruction target**

Notation.

x: vectors
X: matrices
v th row of **X**: **x**v

$\mathcal{G} = (\mathcal{V}, \mathcal{E})$: undirec$\mathcal{V}$ and $\mathcal{E}$ 무방향성 그래프)

: sets of node와 edges

$\text{Each node } v \in \mathcal{V} \text{ has a } F\text{-dimensional attribute vector } \mathbf{x}_v \in \mathbb{R}^F$

if. node 속성을 사용할 수 없는 경우 x를 one-hot index vector또는 learnable parameters로 초기화

https://arxiv.org/pdf/2201.02534v1.pdf

Notation.

x: vectors
X: matrices
v th row of **X**: **x**v

$\mathcal{G} = (\mathcal{V}, \mathcal{E})$:
undirec$\mathcal{V}$ and $\mathcal{E}$ (방향성 그래프)

:
sets of node와 edges

Each node $v \in \mathcal{V}$ has a F-dimensional attribute vector $\mathbf{x}_v \in \mathbb{R}^F$

if. node 속성을 사용할 수 없는 경우 x를 one-hot index vector또는 learnable parameters로 초기화

Masked Input Graph

network masking

- perturb the original graph G by randomly masking a subset of edges $\mathcal{E}_{mask} \cup \mathcal{E}_{reserve} = \mathcal{E}$.
- To make the process more efficient, we adopt random sampling with a high masking ratio to obtain the masked edge set ⇒ 2가지 방법 채택
  - Undirected masking: node v and v사이 link를 무방향취급 $e_{v,u}$ $e_{u,v}$ only one copy is included in $\mathcal{E}$, the masked edge set $\mathcal{E}_{mask}$ is also undirected.
    performing random sampling over
    , $e_{v,u}$ and $e_{u,v}$ are different, and both of them are included in $\mathcal{E}$.
  - Directed masking: Undirected masking 반대 ⇒ resultant masked edge set $\mathcal{E}_{mask}$ is directed.
    performing random sampling
  - 두 masking 기법 차이점은 Undirected masking이 Directed masking보다 재구성을 더 어렵게 만든다는 것
  - 하지만 In network masking scenarios, the best sampling strategy should be related to the network structure
    - if the network is dense, it is better to adopt the tougher strategy (i.e., undirected masking)
    - In contrast, if the original graph is sparse, the directed masking could be the better choice (Undirected masking은 node의 인접 구조를 파괴 ⇒ leading to substantial expressive power degradation of GNN encoders)

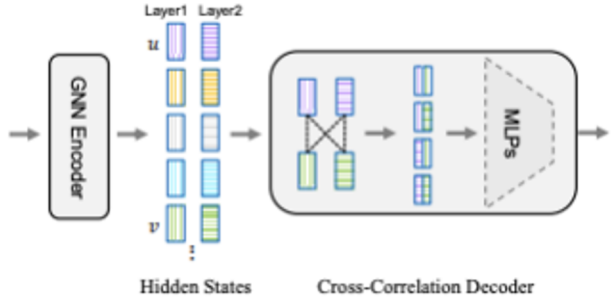
https://arxiv.org/pdf/2201.02534v1.pdf

Notation.

x: vectors
X: matrices
v th row of **X**: **x**v

$\mathcal{G} = (\mathcal{V}, \mathcal{E})$:
undirec$\mathcal{V}$ and $\mathcal{E}$ 무방향성 그래프)

:
sets of node와 edges



GNN Encoder

- follow standard GAE approaches to adopt the well-established GNN ı $\mathbf{h}_v^{(k)} = \mathrm{COM}(\mathbf{h}_v^{(k-1)}, \mathrm{AGG}(\{\mathbf{h}_u^{(k-1)} : u \in \mathcal{N}_v\}))$,
- GCN and GraphSage architectures as backbones
- encoder only operates on a small subset (e.g., 30%) of the full e $\mathcal{E}$ e set for message propagation, $\mathcal{E}_{mask}$ ges in
  는 traing 중에 removed 됨
- 동일한 masking 비율로 training에 사용되는 총 edge 수는 두 샘플링 방법에 대해 동일 (GNN의 message propagation가 기본적으로 bidirectional(양방향))

Cross-correlation decoder

- MGAE decoder는 GNN encoder에 의해 생성된 K-hidden representation matrices {H(1), H(2), · · · , H(K)}를 generated
- 각 node v는 $\{\mathbf{h}_v^{(k)\in\mathbb{R}^d}\}_{k=1}^K$ 표시된 K em $\mathbf{h}_v^{(k)}$ ling $\mathcal{E}_{reserve}$ 가짐, 는 에 의해 정의된 k hops(??)내에서 v의 이웃으로부터의 메시지를 취합하여 얻음
  ı $\mathbf{H}^{(K)}$
- given an edge $e_{v,u}$ and their K hidden representations $\{\mathbf{h}_v^k, \mathbf{h}_v^{(k)}\}_{k=1}^K$, we aim to capture their cross-correlations as below. 다른 granularities로
- $\mathbf{h}_{e_{v,u}} = ||_{k,j=1}^K \mathbf{h}_v^{(k)} \odot \mathbf{h}_u^{(j)}$,
  - || denotes concatenation and ⊙ denotes the element-wise multiplication.
  - $\mathbf{h}_{e_{v,u}} \in \mathbb{R}^{dK^2}$ $e_{v,u}$. $\mathbf{h}_v^{(k)} \odot \mathbf{h}_u^{(j)}$
  - : final representation for edge / : k-th 이웃과 j-th 이웃을 고려하여 node v와 node u 사이의 cross representation

https://arxiv.org/pdf/2201.02534v1.pdf

Notation.

x: vectors
X: matrices
v th row of **X**: **x**v

$\mathcal{G} = (\mathcal{V}, \mathcal{E})$:
undirec$\mathcal{V}$ and $\mathcal{E}$ r방향성 그래프)

:
sets of node와 edges

Each node $v \in \mathcal{V}$ has a F-dimensional attribute vector $\mathbf{x}_v \in \mathbb{R}^F$

if. node 속성을 사용할 수 없는 경우 x를 one-hot index vector또는 learnable parameters로 초기화

Target Graph

Reconstruction target

- MGAE decoder가 masked edge만 재구성하도록 강제함으로써, GNN encoder는 network noise에 강하고 graph 정보를 더 효과적으로
- standard graph-based loss function

$$\mathcal{L} = -\sum_{(v,u)\in\mathcal{E}_{mask}} \log \frac{\exp(\mathbf{y}_{vu})}{\sum_{z\in\mathcal{V}} \exp(\mathbf{y}_{vz})}$$

- $\mathbf{y}_{v,u} = \mathrm{MLP}(\mathbf{h}_{e_{v,u}})$ edge $e_{v,u}$ ructed score for / MLP activation function: R
- 근데.. in experiments에서는 loss fun의 분모의 합계 연산은 계산적으로 불가능 ⇒ adopt negative sampling to accelerate the optimization
- stochastic gradient descent algorit

Algorithm 1: Masked Graph Autoencoder (MGAE)

Input: Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, GNN encoder depth K, masking ratio ω, embedding dimension d;

while not converged **do**
  Randomly masking $\mathcal{G}$ with ratio ω into two edge sets: $\mathcal{E}_{mask}$ and $\mathcal{E}_{reserve}$;
  Perform GNN encoding on the reserved edge set $\mathcal{E}_{reserve}$ according to Eq. (1);
  Obtain the cross representations of edges in $\mathcal{E}_{mask}$ according to Eq. (2);
  Model update by minimizing the reconstruction loss over $\mathcal{E}_{mask}$ according to Eq. (3);
Return The trained MGAE model.

- After training the MGAE model, use the output of decoder to perform link prediction tasks on unseen edges
- For node classification, use the GNN encoder to generate representations for nodes in the graph.
- concatenate K representations of each node as its final representation fed into the classifier.

https://arxiv.org/pdf/2201.02534v1.pdf

Experiments.

- Q1: How effective is MGAE against the state-of-the-art models on link prediction and node classification?
- Q2: How does our model perform under different masking ratios?
- dataset
    - prediction performance: six benchmark graph datasets (three Planetoid datasets (Cora, Citeseer and Pubmed), and three OGB datasets (ogbl-ddi, ogbl-collab, and ogbl-ppa))
    - node classification: five datasets (Cora, Citeseer and Pubmed and two OGB node classification datasets: ogbn-arxiv and ogbn-proteins)

- built upon Pytorch and PyG (PyTorch Geometric) library
    - 200 epochs with Adam optimizer and early stopping with a patience of 50 epochs
    - three hyper-parameters: masking ratio ω, embedding dimension d, and encoder layer K
        - K = 2, ω = 0.7 in default
        - embedding dimension: fix d = 128 and d = 256 for Planetoid and OGB datasets
        - apply undirected masking for Planetoid datasets and directed masking for OGB datasets as default

Table 1: Dataset Statistics.

Data	# Nodes	# Edges	# Features	Split ratio	# Classes
Cora	2,708	5,429	1,433	85/5/15	7
CiteSeer	3,312	4,660	3,703	85/5/15	6
PubMed	19,717	44,338	500	85/5/15	3
ogbl-ddi	4,267	1,334,889	-	80/10/10	–
ogbl-collab	235,868	1,285,465	128	92/4/4	–
ogbl-ppa	576,289	30,326,273	-	70/20/10	–
ogbn-arxiv	169,343	30,326,273	128	–	40
ogbn-proteins	132,534	39,561,252	8	–	112

https://arxiv.org/pdf/2201.02534v1.pdf

Experiments.

- Q1: How effective is MGAE against the **state-of-the-art models on link prediction** and node classification?

Table 2: Link prediction results on Planetoid data with a masking ratio 0.7. The best results are highlighted.

	Cora		CiteSeer		PubMed	
	AUC	AP	AUC	AP	AUC	AP
GCN	37.07 ± 5.07	51.56 ± 4.19	44.75 ± 1.07	52.30 ± 1.01	2.52 ± 0.47	$\mathbf{10.82 \pm 1.04}$
GraphSage	53.90 ± 4.74	65.80 ± 6.94	$\mathbf{54.63 \pm 1.12}$	$\mathbf{60.23 \pm 1.20}$	1.87 ± 0.67	8.92 ± 2.28
ARGE	20.43 ± 4.66	23.86 ± 6.53	28.39 ± 2.51	37.66 ± 1.98	0.41 ± 0.26	3.83 ± 0.84
SelfTask-GNN	42.26 ± 4.85	50.67 ± 6.02	33.03 ± 5.11	42.22 ± 7.14	0.65 ± 0.30	4.26 ± 1.27
MGAE-GCN	$\mathbf{65.91 \pm 3.50}$	$\mathbf{75.02 \pm 2.26}$	54.74 ± 1.06	61.01 ± 1.18	$\mathbf{3.98 \pm 1.33}$	9.97 ± 1.55
MGAE-SAGE	$\mathbf{66.00 \pm 9.49}$	$\mathbf{75.18 \pm 6.57}$	49.27 ± 0.96	55.44 ± 0.82	1.37 ± 0.38	4.79 ± 0.16

- Our model MGAE performs better than graph autoencoder based baselines (AGRE, GAE, and GraphSage) on six datasets in almost all cases. Specifically, MGAE achieves comparable results with the best graph autoencoder baselines on ogbl-collab and ogbl-ppa datasets, while significantly outperforms them on other four datasets with great margin. In six datasets, MGAE obtains new state-of-the-art performance on Cora, PubMed and ogb-ddi datasets.
- Compared with self-supervised learning baselines (DGI, GIC, and SelfTask-GNN), our model MGAE achieves substantial performance gains on five out of six datasets. In particular, MGAE only loses to GIC on CiteSeer dataset while outperforms on other five datasets. Besides, we also observe that the performance gap between our model and three self-supervised baselines increases on OGB datasets. This result shows that graph autoencoder architecture is more suitable for link prediction task on large-scale datasets.
- Another important observation is that *none* of our two variants MGAE-GCN and MGAE-SAGE can consistently outperforms the other in six datasets. For example, although GAE performs better than GraphSage on Cora and CiteSeer datasets, MGAE-SAGE outperforms MGAE-GCN on these two datasets. It indicates that the best GNN encoder for MGAE varies on different graph scenarios.

⇒ GAE models

⇒ self-supervised methods

Planetoid datasets (Cora, CiteSeer, and PubMed)

- randomly split all edges into three sets
- i.e., the training set (85%), the validation set (5%), and the test set (10%)

- Another promising property of MGAE we want to remark is that the results in Table 2 and 3 are obtained under a high masking ratio ($\omega = 0.7$), excepting ogbl-ppa. That is, we only feed 30% original edges of the original graph to GNN encoder. Therefore, MGAE is naturally more efficient than traditional graph autoencoder models, since message propagation is the most time-consuming process of GNN models. Besides, it also indicates that a lot of node connections in graph data are redundant. This observation is consistent with the motivations for structure learning [31, 32] or graph sparsification [33, 34].

For OGB datasets (ogbl-ddi, ogbl-collab, and ogbl-ppa)

- split the datasets into three sets according to the split ratios summarized in Table 1
- evaluate their performance using Hit rate (Hits@N), where N is the number of nodes recalled.

https://arxiv.org/pdf/2201.02534v1.pdf

Experiments.

- Q1: How effective is MGAE against the **state-of-the-art models on** link prediction and **node classification**?

Planetoid (Cora, CiteSeer, and PubMed) and OGB (ogbn-arxiv and ogbn-protiens) datasets

Table 4: Node classification performance on all datasets based on 70% randomly edge masking.

Method	Cora ACC.	CiteSeer ACC.	PubMed ACC.	ogbn-arxiv ACC.	ogbn-proteins AUC
GCN	83.60 ± 0.52	63.37 ± 1.21	78.23 ± 1.63	66.01 ± 0.37	61.67 ± 0.35
GraphSage	74.30 ± 1.84	60.20 ± 2.15	81.96 ± 0.74	64.79 ± 2.91	55.39 ± 0.79
ARGVA	85.86 ± 0.72	73.10 ± 0.86	81.85 ± 1.01	50.06 ± 1.21	40.73 ± 0.68
SelfTask-GNN	84.69 ± 0.09	71.82 ± 0.13	83.92 ± 0.18	68.30 ± 0.02	60.93 ± 0.44
DGI	85.41 ± 0.34	74.51 ± 0.51	85.95 ± 0.66	67.08 ± 0.43	50.31 ± 0.55
GIC	$\mathbf{87.70 \pm 0.01}$	$\mathbf{76.39 \pm 0.02}$	85.99 ± 0.13	64.00 ± 0.22	48.55 ± 0.47
MGAE	86.15 ± 0.25	74.60 ± 0.06	$\mathbf{86.91 \pm 0.28}$	$\mathbf{72.02 \pm 0.05}$	$\mathbf{63.33 \pm 0.12}$

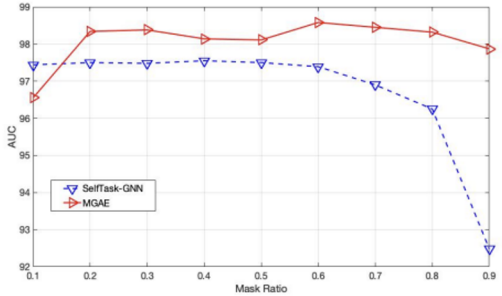
- randomly split 10% of all edges into validation set and use the remained 90% edges as training set
- The validation set is used to tune hyperparameters
- After the model is trained, use the full set of edges as input to generate node representations for downstream evaluation
- train a SVM classifier on the learned node representations of all models, and apply 5-fold cross-validation to estimate the performance
- To avoid randomness, we repeat the process for 10 times and report the average result in terms of Accuracy (ACC.) for Core, CiteSeer, PubMed and ogbn-arxiv and AUC for ogbn-proteins

- Our model MGAE performs consistently better than three graph-autoencoder based baselines (GCN, GraphSage, and ARGVA) across five datasets. Given that MGAE can obtain at least comparable results with classical graph autoencoders in link prediction task, it indicates that the proposed MGAE is a powerful graph autoencoder alternative.
- Compared with self-supervised learning based models (DGI, GIC, and SelfTask-GNN), our model loses to the best performance of them on two small datasets (Cora and CiteSeer). However, MGAE outperforms them on three large datasets (PubMed, obgn-arxiv, and ogbn-proteins) by a large margin.
- SelfTask-GNN is a closely related work to MGAE, since they all focus on randomly masking some edges as self-supervised training task. However, according to the results in Table 2, 3, and 4, SelfTask-GNN does not perform well compared to state-of-the-art baselines due to the limited decoder design and masking strategies. These results show that our MGAE model is the first work that can successfully adopt self-supervised learning to boost the performance of classical graph autoencoders.

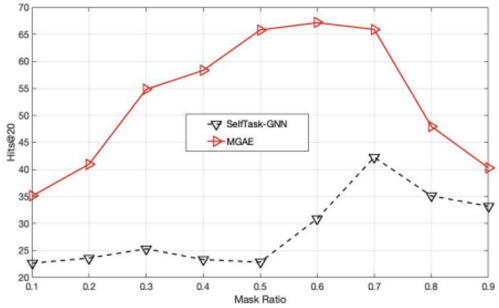
Experiments.

- Q2: How does our model perform under different masking ratios?

the results of MGAE-GCN and SelfTask-GNN on PubMed and ogbl-ddi datastes.



(a) PubMed



(b) ogbl-ddi

- The performance of MGAE first increases with the increasing of masking ratio ω until it reaches 0.7, then it drops when ω further increases. Besides, our MGAE model performs relatively stable when ω is around 0.5 to 0.7. These results indicate the robustness of our model under high masking ratios.
- MGAE performs consistently better than SelfTask-GNN on the two datasets across different ω values, except when $\omega < 0.2$ on PubMed dataset. Besides, the performance gap is more significant when ω is around 0.5 and 0.7. These observations verify the effectiveness of our proposed cross-correlation decoder for self-supervised graph autoencoder training.

End of the Document