Does Negative Sampling Matter? a Review With Insights Into its Theory and Applications

Zhen Yang, Ming Ding, Tinglin Huang, Yukuo Cen, Junshuai Song, Bin Xu, *Member, IEEE*, Yuxiao Dong, and Jie Tang, *Fellow, IEEE*

***Abstract*—Negative sampling has swiftly risen to prominence as a focal point of research, with wide-ranging applications spanning machine learning, computer vision, natural language processing, data mining, and recommender systems. This surge in interest prompts us to question the fundamental impact of negative sampling: Does negative sampling really matter? Is there a general framework that can incorporate all negative sampling methods? In what fields is it applied? Addressing these questions, we propose a general framework that using negative sampling. Delving into the history of negative sampling, we chart its evolution across five distinct trajectories. We dissect and categorize the strategies used to select negative sample candidates, detailing global, local, mini-batch, hop, and memory-based approaches. Our comprehensive review extends to an analysis of current negative sampling methodologies, systematically grouping them into five classifications: static, hard, GAN-based, Auxiliary-based, and In-batch. Beyond detailed categorization, we explore the practical application of negative sampling across various fields. Finally, we briefly discuss open problems and future directions for negative sampling.**

***Index Terms*—Negative sampling algorithms, negative sampling applications, negative sampling framework.**

I. Introduction

DOES *negative sampling matter?* Negative sampling (NS) is a technique used in machine learning, which aims to select a small subset of negative samples to replace the entire pool of possible negative samples. Take word2vec [1] as an example, the training objective of the Skip-Gram model in word2vec is designed to maximize the likelihood of predicting context words within a certain range before and after the target word. Mathematically, for a sequence of training words $w_1, w_2, \ldots, w_T$, the objective is to maximize the average log probability, which can be presented as:

$$\min -\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

where T is the length of the word sequence, c is the size of the training context (that is a window of words around the target word), w_t is the target word, w_{t+j} are the context words within the window around w_t.

The $p(w_{t+j}|w_t)$ is the probability of observing a context word w_{t+j} given a target word w_t, defined by the softmax function:

$$p(w_{t+j}|w_t) = \frac{exp({v'_{w_{t+j}}}^T v_{w_t})}{\sum_{w=1}^{W} exp({v'_w}^T v_{w_t})}$$

where v_w and v'_w are the "input" and "output" vector embeddings of word w, and W is the number of words in the vocabulary.

The goal of training is to learn the word vectors v_w and v'_w such that this probability is maximized for the actual context words and minimized for other words. In practice, the Skip-Gram model with the standard softmax function is computationally expensive, especially for large vocabularies. Each training step requires calculating and normalizing the probabilities of all words in the vocabulary given a target word, which is computationally intensive.

Negative sampling is leveraged to simplify this. Instead of using all words in the vocabulary, the model only considers a small subset of negative words (not present in the current context) along with the actual positive context words. This substantially reduces the computational burden. Besides, negative sampling changes the training objective. Instead of predicting the probability distribution of the entire vocabulary for a given input word, it reformalizes the problem as a binary classification task. For each pair of words, the model predicts whether they are likely to appear in the same context (positive samples) or not (negative samples). The objective of the Skip-Gram model with negative sampling can be represented as:

$$\min -\log\sigma({v'_{w_{t+j}}}^T v_{w_t}) - \sum_{i=1}^{k}\mathbb{E}_{w_i\sim p_n(w)}[\log\sigma(-{v'_{w_i}}^T v_{w_t})]$$

where k is the number of negative samples, w_i are negative words sampled for the target word w_t, $\sigma(\cdot)$ is the sigmoid function and $p_n(w)$ is the negative sampling distribution that is designed as the 3/4 power of word frequency.

Manuscript received 7 February 2023; revised 21 November 2023; accepted 12 February 2024. Date of publication 29 February 2024; date of current version 2 July 2024. This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFF1201300, in part by the NSFC for Distinguished Young Scholar under Grant 61825602 and in part by the New Cornerstone Science Foundation through the XPLORER PRIZE. Recommended for acceptance by M. Kloft. *(Corresponding authors: Bin Xu; Jie Tang.)*

Zhen Yang, Ming Ding, and Yukuo Cen are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: yangz21@mails.tsinghua.edu.cn; dm18@mails.tsinghua.edu.cn; cyk20@mails.tsinghua.edu.cn).

Tinglin Huang is with the Department of Computer Science, Yale University, New Haven, CT 06511 USA (e-mail: tinglin.huang@yale.edu).

Junshuai Song is with the Technology and Engineering Group, Tencent, Beijing 100080, China (e-mail: jasonjssong@tencent.com).

Bin Xu, Yuxiao Dong, and Jie Tang are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: xubin@tsinghua.edu.cn; yuxiaod@tsinghua.edu.cn; jietang@tsinghua.edu.cn).

Digital Object Identifier 10.1109/TPAMI.2024.3371473

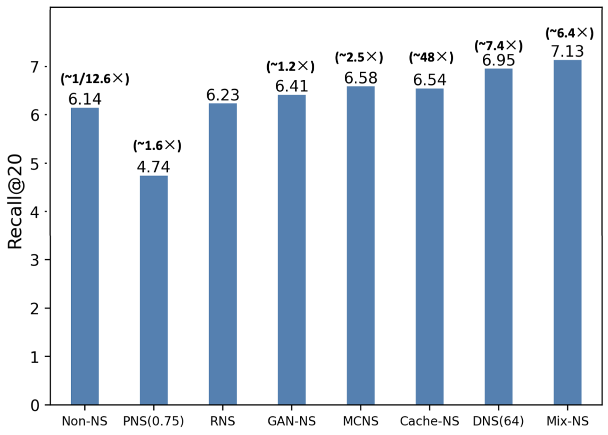


Fig. 1. Performance and Convergence speed comparison in the RS domain with the Non-NS and other methods using negative sampling on the Yelp2018 datasets with LightGCN as an encoder. PNS(α) denotes popularity-based negative sampling with various power parameters α. DNS(β) represents dynamic negative sampling with a different number of negative sample candidates β. The details of the negative sampling methods can be found in Section II.

This example demonstrates the importance of negative sampling. Without it, the model would need to compute the co-occurrence probability with every other word for each training instance, which is computationally expensive, particularly for large vocabularies. By employing negative sampling, word2vec significantly reduces the computational burden and achieves great performance.

In a broader perspective, negative sampling is a widely-used and valuable technique technique across various domains, such as recommendation systems (RS) [2], graph representation learning (GRL) [3], [4], [5], [6], [7], knowledge graph learning (KGE) [8], [9], natural language processing (NLP) [1], [10], [11], [12], and computer vision (CV) [13], [14], [15], [16], [17], [18]. In each of these domains, the core principle of negative sampling remains the same – selecting a representative subset of negative samples to improve the efficiency and effectiveness of the learning process.

To clearly verify the improvements, we compare a basic method with other negative sampling methods on performance and convergence speed in recommendation system. As shown in Fig. 1, compared with the basic RNS, the method without negative sampling (Non-NS) achieves a close performance but needs a longer training time since it uses all negatives for model training. Compared to RNS, other advanced negative sampling methods can reduce the computational burden, accelerate training convergence(*up to* $\sim 48\times$ with Cache-NS), degrade performance bias, and boost performance (*14% improvement* with Mix-NS). Compared with RNS, PNS (0.75) achieves a faster convergence of $1.6\times$ but reduces performance by 24%. Compared with Mix-NS, DNS (64) shows faster convergence but slightly degrades performance.

However, the specific implementation of negative sampling can vary significantly. Is there a general framework that can incorporate all the negative sampling methods and be applicable to different domains? In this survey, we formalize negative sampling and propose a general framework that uses negative sampling for model training (See Fig. 2). Under this framework, we categorize different negative sampling methods to make them more accessible and comparable. Besides, we summarize three critical aspects to be taken into account when designing a better negative sampling method: (1) where to sample from? (2) how to sample? and (3) in which field should it be applied?

Contributions: The main contributions of this survey are presented as follows:

- We highlight the importance of negative sampling and propose a general framework that can incorporate existing negative sampling methods from various domains.
- We identify three aspects that should be considered for designing negative sampling: negative sample candidates, negative sampling distributions, and negative sampling applications. We sum up five selection methods for negative sample candidates and categorize existing negative sampling methods into five groups.
- We report the characteristics of negative sampling methods in various domains and demonstrate the pros and cons of each type of negative sampling method. We summarize several open problems and discuss the future directions for negative sampling.

Survey Organization: The rest of this survey is organized as follows. In Section II, we provide the overview of negative sampling, first briefly tracing back the development history of negative sampling and then giving a general framework for negative sampling. Section III reviews existing negative sampling algorithms, as well as the pros and cons of each category. Section IV further explores applications of negative sampling in various domains. Finally, we discuss the open problems, future directions, and our conclusions in Sections V and VI.

Variables Definitions: Here, we elucidate the meanings of the variables employed in our survey. x, x^+, and x^- denote an anchor sample, a positive sample, and the selected negative sample respectively. p_n represents a designed negative sampling distribution. x' denotes a negative candidate within the pool of negative sample candidates $\mathcal{C}$. These candidates are the potential selections for the negative sample x^-, as determined by the negative sampling strategy. $S(\cdot)$ is a function used to measure the similarity between samples, such as dot product [6], [19], L1 and L2 norms [20], [21], depending on the specific requirements of the learning task. $f(\cdot)$ represents the model that maps these input samples into their respective embeddings.

II. Negative Sampling

In this section, we provide a comprehensive overview of negative sampling, detailing its formalization and framework that uses negative sampling for model training, tracing its historical development from five development lines, and elaborating its important role in machine learning.

A. Formalization and Framework

Negative sampling aims to select a subset of negative samples from a larger pool, which is a technique used to improve the efficiency and effectiveness of training in machine learning

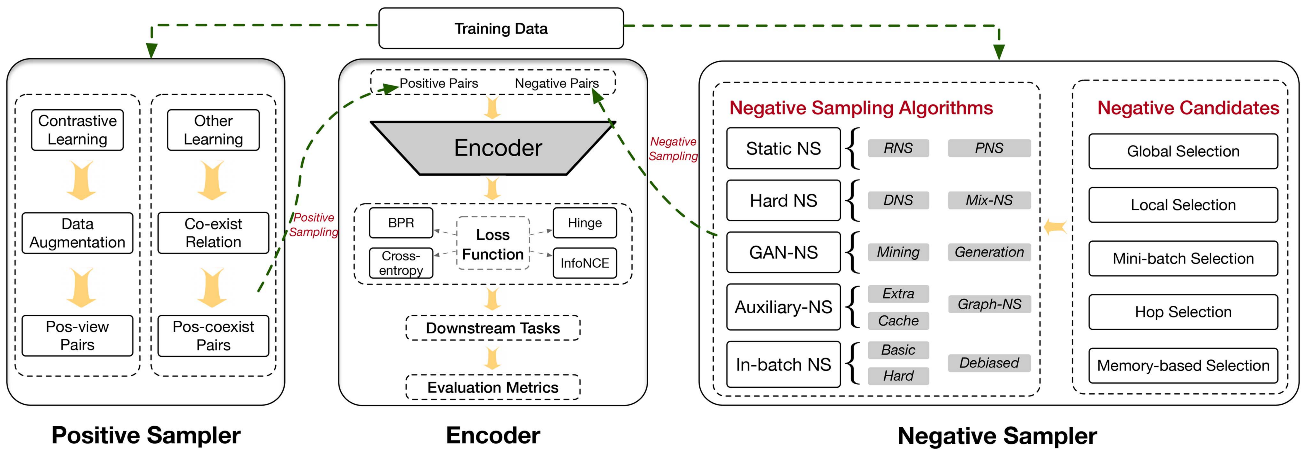


Fig. 2. Illustration of a general framework that uses negative sampling. Positive and negative pairs are sampled implicitly or explicitly by positive and negative samplers respectively, both of them composing the training data. An encoder is applied for latent representation learning in various domains. In contrastive learning, positive pairs (i.e., Pos-view Pairs) are derived from data augmentations of the same instance or different perspectives of the same entity, while in other domains, such as metric learning, positive pairs (Pos-coexist Pairs) are the other instances in the dataset.

models. The formalization of negative sampling can be encapsulated in a general loss function. The loss function incorporating negative sampling typically maximizes the similarity of positive pairs while minimizing the similarity of negative pairs. For a specific positive sample x^+ or anchor sample x, negative sampling selects negative sample x^- from the pool of negative sample candidates $\mathcal{C}$ based on a negative sampling distribution $p_n(x^-)$ for model training.

$$\mathcal{L} = l(x, x^+, x^-), x^- \sim p_n(x^-)$$

where $l(\cdot)$ is a specific loss function in various domains.

In different application domains, the specific loss function may vary based on various tasks and datasets.

- *Bayesian Personalized Ranking (BPR) Loss for Pair-based Sampling:* For a pair of positive and negative samples (pos, neg), BPR loss focuses on ensuring that the log-likelihood of a positive sample is higher than that for a negative sample.
- *Hinge Loss for Triplet-based Sampling:* For a triplet of an anchor, a positive sample, and a negative sample (anchor, pos, neg), hinge loss ensures that the similarity between the anchor and the negative sample is lower than that between the anchor and the positive sample by at least a pre-set margin.
- *Cross-Entropy Loss for Single Positive with Multiple Negatives:* In cases with one positive and multiple negatives, cross-entropy loss is used to discriminate the positive sample from a set of negatives.
- *InfoNCE Loss in Contrastive Learning:* InfoNCE loss, common in contrastive learning frameworks, is designed to contrast positive pairs against negative pairs.

These loss functions share a unifying principle of drawing positive pairs closer and distancing negative pairs in the model's representation space. Detailed formulations of these functions can be found in Table I. Based on the formalization of negative sampling, we propose a general framework that uses negative sampling for model training, which contains a positive sampler, a negative sampler, and a trainable encoder. The overall framework is illustrated in Fig. 2. The positive sampler is applied

TABLE I
Overview of Loss Functions Used in a General Framework That Uses Negative Sampling for Model Training

Loss Formulation
$\mathcal{L}_{\text{BPR}} = \ln\sigma(f(x^+) - f(x^-))$
$\mathcal{L}_{\text{Hinge}} = \max(0, S(f(x), f(x^+)) - S(f(x), f(x^-)) + \gamma)$
$\mathcal{L}_{\text{CE}} = -[\log\sigma(S(f(x), f(x^+))) + \log(\sigma(-S(f(x), f(x^-)))]$
$\mathcal{L}_{\text{InfoNCE}} = -\log\frac{exp(S(f(x),f(x^+))/\tau)}{exp(S(f(x),f(x^+))/\tau)+\sum_{k=1}^{B-1}exp(S(f(x),f(x_k^-))/\tau)}$

τ represents a temperature parameter that scales the similarity scores. B denotes the batch size.

to generate positive training pairs. For example, positive pairs in recommendation are sampled explicitly from the observed user-item interactions, while these in contrastive learning are generated implicitly by data augmentations. Negative training pairs are sampled by different negative sampling strategies via a negative sampler. The abovementioned sampled pairs serve as training data and are fed into a trainable encoder. The trainable encoder varies by application domains, such as graph neural networks (GNNs) [22], [23], [24] in graph representation learning, ResNet [25] in unsupervised visual representation learning, BERT [26] and RoBERTa [27] in unsupervised sentence embedding learning, Skip-Gram [1] in word embedding, TransE [20] and TransH [28] in knowledge graph embedding.

B. Negative Sampling History

The history of negative sampling is a fascinating development line through the evolution of machine learning techniques. Here, we delve into the historical development of negative sampling, highlighting the motivations behind various negative sampling methods. As shown in Fig. 3, the earliest implementation of negative sampling is random negative sampling (RNS). The basic idea of RNS is to randomly select a subset of negative samples from the large pool, thereby reducing the computational requirements of the training process. Notably in 2008, Pan et al. [29] utilized RNS to prevent the model from developing a bias towards the majority of negative samples in one-class

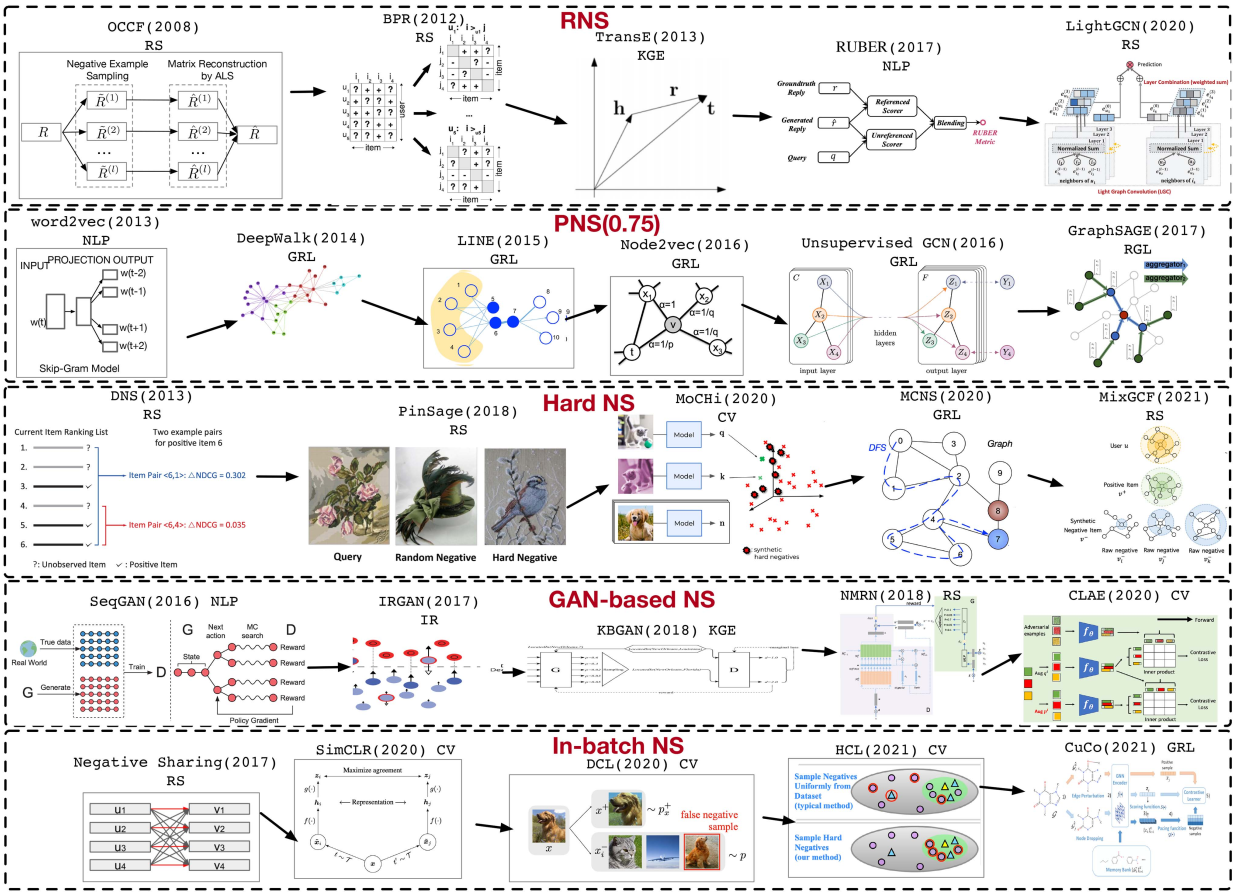


Fig. 3. Five development lines of negative sampling. Each development line addresses different challenges and has been adopted and adapted to suit the specific needs of individual domains, such as RS, NLP, GRL, KGE, and CV.

collaborative filtering (OCCF) for recommender systems, which effectively balanced the data used in OCCF. Rendle et al. [2] proposed a pair-wise learning method, which leveraged pair-wise loss and random negative sampling to solve the lack of negative data in implicit feedback. Such a random negative sampling method is also applied to graph embedding [20], [30] and GNN-based recommendation [31], [32].

Over time, as datasets grew and models became more complex, the limitations of RNS became apparent. These randomly selected negative samples might not always be relevant for the model training. Some works [1], [3], [4], [5], [24] believe that sampling negative samples based on attributes of the training dataset is a more refined method. Thus, popularity-based negative sampling (PNS) is proposed to select negative samples based on their frequency or popularity or degree, operating under the assumption that frequently occurring negative samples are more likely to be true negatives. This approach aims to make the negative sampling more meaningful and tailored to the specific dataset. A commonly-used approach in PNS is to sample negatives in a way that is proportional to the 3/4 power of their frequency. This particular distribution is found to be effective in various domains, such as natural language processing [1] and graph embedding learning [3], [5], [24].

However, the selected negative samples in RNS and PNS are static and not adaptive to the model training, which can lead to suboptimal performance. To address this, hard negative sampling strategies (Hard NS) [19], [33], [34], [35] are proposed to select negatives that are difficult for the model to distinguish correctly, thereby providing a more effective learning signal. This approach aligns the negative sampling more closely with the model's current learning process, leading to more efficient training. These hard negatives provide more information for gradients in model optimization, which can accelerate convergence and boost performance. Hard NS is applied to word embedding [36], answer selection [37], knowledge graph embedding [38], graph embedding learning [6], GNN-based recommendation [7], [39], [40], and object detecion [41].

Inspired by GAN [42], a number of works [9], [43], [44], [45], [46], [47] propose an adversarial framework that takes advantage of adversarial training and adversarial examples to generate negative samples. By using a GAN, GAN-based negative sampling (GAN-based NS) can generate negative samples that are more realistic and representative of true negative samples, which might not be readily available in the dataset. Here, the generator serves as a negative sampler while the training model acts as the discriminator. Such methods rely on a generator that adaptively approximates the underlying negative sampling distribution. GAN-based NS is a general strategy that can be applied to recommendation, graph learning, knowledge graph learning, and unsupervised visual representation learning.

Recently, contrastive learning has achieved tremendous success in unsupervised visual representation learning. In-batch negative sampling (In-batch NS) is a specific strategy used in contrastive learning that effectively leverages other samples within the same batch as negative samples. This method is computationally efficient as it does not require additional steps to generate or select negative samples. For example, SimCLR [48] utilizes other samples in the current mini-batch as negatives. MoCo [49] maintains a memory bank to store the past several batches as negative samples for model training. One limitation of In-batch NS is that the quality of negatives is constrained by the batch size. Smaller batches may not provide a sufficiently varied set of negatives, potentially impacting the model's ability to learn finer representations. To overcome this limitation, several works [16], [17], [18], [50], [51], [52] have focused on introducing hard negatives into the contrastive learning paradigm.

C. The Importance of Negative Sampling

Here, we explicitly elaborate on the *importance* of negative sampling from three aspects.

1) *Computational Efficiency:* By selecting a representative subset of negative samples, negative sampling eliminates the need for the model to consider all possible negative samples during training. In original learning without negative sampling, the model often involves predicting the probability distribution of a given sample. This requires computing probabilities between a given sample and all samples in the dataset, which is computationally intensive, especially for large datasets [1], [29]. Negative sampling simplifies this by transforming the task into a binary classification problem. Instead of predicting the probability distribution across the entire dataset, the model learns to differentiate between positive samples and a small number of selected negative samples. Thus, negative sampling significantly reduces the computational burden and accelerates the training process.
2) *Handing Class Imbalance:* In real-world datasets, there is often a significant imbalance between the positive and negative samples, which can lead to performance bias. Negative sampling addresses this issue by ensuring that the training process is exposed to a more balanced dataset. By selecting a subset of negative samples, it prevents the model from developing a bias towards the majority class, thereby improving the model's ability to accurately predict less frequent classes. In cases like one-class application scenarios that only target positive samples with an inherent limitation of the absence of explicit negative feedback, the challenge is even more pronounced. Compared to the approach of treating all non-positive samples as negatives, negative sampling selectively chooses a representative subset of these non-positive samples as negatives. Such a refined method provides more balanced data for the model training. Thus, negative sampling is a key technique in addressing performance bias in machine learning models dealing with real-world data.
3) *Improved Model Performance:* Negative sampling helps in improving model performance by learning more meaningful representations. By focusing on a subset of more informative negative samples, the model can better capture the subtle distinctions between different samples. As highlighted in studies [6], [13], [14], [53], [54], negative samples, particularly more informative or "hard" negatives, contribute significantly to the gradients during the training process. *Hard negatives* are those samples that are similar to positive samples in feature space, making the difficult for the model to distinguish from the positives. Training with these hard negative samples forces the model to learn finer distinctions because these negatives contribute more significantly to the gradients, leading to a more effective optimization process and improvement in the model's ability to distinguish between positive and negative samples. Thus, negative sampling, especially when it involves the strategic use of hard negatives, is an essential technique for improving the performance of machine learning models.

III. Algorithms

In this section, we first summarize five categories of selection methods to form negative sample candidates (i.e., where to sample from?). Next, we summarize a variety of negative sampling algorithms into five categories (i.e., how to sample?).

A. Negative Sample Candidates Construction

The initial step of negative sample candidate construction is foundational, where the pool of potential negative samples $\mathcal{C}$ is established. Here, we answer the first question where should we sample negative examples from? In terms of the composition of the negative sample candidates, we summarize their selection methods into the following five categories.

- *Global Selection* is one of the most common methods for negative sample candidate selection, where the pool of negative samples is composed of all possible negatives from the entire dataset. It ensures diversity in the negative samples but may include less relevant negatives, which could impact the learning efficiency. For example, word2vec [1] utilizes the whole vocabulary as the pool of possible negative samples; LightGCN [32] leverages all unobserved items as the pool.
- *Local Selection* focuses on sampling a specific subset of the total available negatives as the pool of negative samples. This method is more selective compared to Global Selection, aiming to construct a pool that is more relevant or challenging for a specific query or anchor. For example, ANCE [104] selects top-k negative samples based on the query as the pool of negative samples. Besides, the specific subset can reduce the computational complexity that would be involved in handling the entire set of available negatives. For example, DNS [19] randomly samples a subset as the pool of negatives for probability calculations.
- *Mini-batch Selection* uses other samples in the current mini-batch as the pool without the additional process of choosing. It leverages the data already loaded into memory for the batch, making it computationally efficient. For

TABLE II
OVERVIEW OF NEGATIVE SAMPLING METHODS COLLECTED FROM VARIOUS DOMAINS

Cate	Subcategory	Model Recipe	Candidates
S	PNS	Word2Vec [1]$^{(NLP)}$, Deepwalk [3]$^{(GRL)}$ LINE [4]$^{(GRL)}$,Node2vec [5]$^{(GRL)}$	Global
	RNS	BPR [2]$^{(RS)}$, LightGCN [32]$^{(RS)}$ TransE [20]$^{(KGE)}$, DISTMULT [55]$^{(KGE)}$,RUBER [56]$^{(NLP)}$,USR [57]$^{(NLP)}$	
H	DNS	FaceNet [13]$^{(CV)}$, Max Sampling [37]$^{(NLP)}$, PinSage [39]$^{(RS)}$, [41], [58], [59], [60] SGA [36]$^{(NLP)}$,DNS [19],AOBPR [61]$^{(RS)}$,BootEA [38],Dual-AMN [62]$^{(KGE)}$ NSCaching [63]$^{(KGE)}$,ESimCSE [12]$^{(NLP)}$,MoCoSE [64]$^{(NLP)}$,MocoRing [18]$^{(CV)}$	Global Local Memory-based
	Mix-NS	MoChi [16]$^{(CV)}$ MixGCF [7]$^{(RS)}$, MixKG [65]$^{(KGE)}$	Cache Global
G	Mining	IRGAN [43]$^{(IR)}$,SeqGAN [66]$^{(NLP)}$,ACE [67]$^{(NLP)}$ KBGAN [9]$^{(KGE)}$,IGAN [68]$^{(KGE)}$, NMRN [45]$^{(RS)}$ GraphGAN [44]$^{(GEL)}$,ProGAN [69]$^{(GRL)}$	Global
	Generation	CFGAN [46]$^{(RS)}$,AdvIR [47]$^{(IR)}$,HeGAN [70]$^{(GRL)}$,SAN [71]$^{(NLP)}$ NDA-GAN [72],AdCo [50], CLAE [73],NEGCUT [74],DAML [75]$^{(CV)}$	
A	Graph	MCNS [6]$^{(GRL)}$,SANS [76]$^{(KGE)}$ GNEG [11]$^{(NLP)}$,SamWalker [77],KGPolicy [40], DSKReG [78], MixGCF [7]$^{(RS)}$	Hop Global
	Extra	SBPR [79]$^{(RS)}$,PRFMC [80]$^{(RS)}$,MF-BPR [81]$^{(RS)}$, View-aware NS [82]$^{(RS)}$ ReinforcedNS [83],RecNS [84]	Local
	Cache	Unsupervised Feature Learning [85]$^{(CV)}$,NSCaching [63]$^{(KGE)}$,SRNS [86]$^{(RS)}$ MoCo [49], MoCo-V2 [87],MoCoRing [18]$^{(CV)}$,GCC [88]$^{(GRL)}$,ESimCSE [12],MoCoSE [64]$^{(NLP)}$	Cache
B	Basic	N.S. [89],S^3-Rec [90],SGL [91],MHCN [92],DHCN [93] $^{(RS)}$, SimCLR [48]$^{(CV)}$ MVGRL [94],GRACE [95],GraphCL [96]$^{(GRL)}$,SimCSE [97],InfoWord [98] $^{(NLP)}$	Mini-batch
	Debiased	DCL [15]$^{(CV)}$,GDCL [99]$^{(GRL)}$	
	Hard	MoCoRing [18]$^{(CV)}$,CuCo [100],ProGCL [101]$^{(GRL)}$,VaSCL [102],SNCSE [103] $^{(NLP)}$	

For acronyms used, "S" represents static NS; "H" refers to Hard NS; "G" means GAN-based NS; "A" means auxiliary-based NS; "B" represents in-batch NS.

example, SimCLR [48] and SimCSE [97] leverage other samples in the same batch as the pool of negatives.

- *Hop Selection* is a novel selection method for graph-structured data, which selects k-hop neighbors as negative sample candidates. This method effectively takes advantage of the graph structure where the information propagation mechanism provides theoretical support. However, matrix operation for obtaining k-hop neighbors is impractical for web-scale datasets. Therefore, a path obtained from a random walk or DFS is often considered a negative sample candidate. For example, RecNS [84] selects the intermediate region (i.e., k-hop neighbors) as the pool of negatives for graph-based recommendation; SANS [76] utilizes k-hop neighbors as the pool for knowledge graph embedding.
- *Memory-based Selection* maintains a memory bank or a cache as a pool to store the pool of negative sample candidates. This bank can retain a large number of negatives from past iterations or batches, which are then used for subsequent training steps. The memory bank is typically updated continuously, with new negatives being added and the oldest ones being removed, ensuring a fresh and diverse set of negative samples. Memory-based Selection is particularly useful in situations where it is beneficial to have a large pool of negative samples to compare against positive samples, such as in contrastive learning applications. For example, MoCo [49] uses a queue bank to store the pool of negative samples.

B. Negative Sampling Algorithms

Once the negative candidates are constructed, the next step involves negative sampling algorithms, which aim to design a negative distribution (i.e., a specific probability distribution) or a sophisticated negative sampling strategy to sample negative samples from the pool of negative candidates for model training.

The design of negative sampling algorithms can be simple, such as random sampling, or it can be more sophisticated, taking into account factors like the current state of the model, the difficulty level of the negatives, or their frequency of occurrence in the dataset. Here, we briefly present an overview of negative sampling algorithms. Table II summarizes existing negative sampling algorithms. The general categorization of negative sampling algorithms and the abbreviated description of each category can be illustrated as follows:

- *Static NS* is a static negative sampling method that keeps a fixed negative sampling distribution during the training process. As a basic negative sampling method, static NS is usually utilized to estimate the performance of a newly proposed model optimized with negative sampling. In general, static NS is a simple, fast, and easy-implemented but model-independent method.
- *Hard NS* is a model-dependent negative sampling method in which the designed probability of selecting a specific negative sample is related to its relevance or difficulty. Here, hard negatives are close to the decision boundary in the feature space, which makes it difficult for the model to distinguish from positive samples. In summary, Hard NS

is a dynamic and mutually promoting method where the applied negative sampling distribution varies with model training.

- *GAN-based NS* is an adversarial negative sampling method that utilizes a generative adversarial network (GAN) [42] to sample negatives. Here, the generator serves as a negative sampler to select or generate negatives (fake positives) to confuse the discriminator while a trainable encoder usually acts as the discriminator to distinguish positive and negative samples. The training process of GAN-based NS is similar to GAN to conduct a minimax game.
- *Auxiliary-based NS* utilizes some auxiliary information to sample negatives, such as extra data, graphs, and cache. Extra-data-based NS incorporates other types of data into negative sampling, such as view data and exposure data collected from E-commerce platforms. Graph-based NS takes advantage of graph information to sample negatives. Cache-based NS maintains a cache to store important candidate negatives with a fixed frequency update mechanism.
- *In-batch NS* is a common negative sampling method that can boost training efficiency when a large number of negatives are required, which allows for the sharing of negative samples in a batch. The number of negatives N for each positive pair is related to the batch size B. In dense retrieval domain, the number of negatives N can be formulated as $N = B - 1$ while can be formulated as $N = 2(B-1)$ in contrastive learning. Besides, it cannot be ignored that the false negative issue has a huge impact on performance for In-batch NS.

Next, we successively review each category of negative sampling algorithms in detail and demonstrate their pros and cons.

1) Static Negative Sampling: Static negative sampling provides a simple and fast negative sampling method and is usually used as a basic baseline, comprising of random negative sampling (RNS) and popularity-based negative sampling (PNS). The general distribution can be represented as: $p_n(x^-) = (\frac{\#x^-}{\sum_{x' \in \mathcal{C}} \#x'})^\beta$ where β is a hyperparameter that controls the sampling distribution, $\#x'$ is the frequency of sample x'.

- *Random Negative Sampling (RNS):* As the most prevalent negative sampling method, RNS is widely used as the default sampling method to evaluate the effectiveness of the proposed model in optimization with negative sampling. RNS designs uniform sampling weights for each negative sample when deciding which ones to use during training. Uniform weights mean that every negative sample has an equal chance of being selected, implying no preference among the negatives. Since RNS uses the same weights for negative samples, it cannot select useful and informative negative samples, leading to a sub-optimal result.
- *Popularity-based Negative Sampling (PNS):* As proposed in previous work [29], user-oriented and item-oriented sampling methods substitute uniform sampling by counting the number of interactions of users and items, respectively. It exploits data properties to sample negatives in early recommendation efforts. Word2vec [1] empirically sets the negative sampling distribution as the $3/4$ power of word frequency for faster training and better word embedding learning. Most later studies on network embedding [3], [4], [5] or graph representation learning [24] also keep this setting for negative sampling. However, the most appropriate choice of β varies with different datasets and fields (See Fig. 4). Results in [11], [54] demonstrate that performance shows a strong dependency on the choice of β.

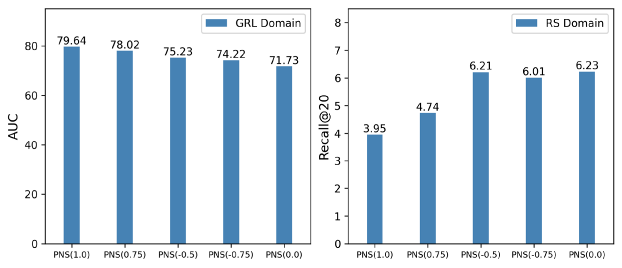


Fig. 4. Performance comparison with different choices of β on GRL and RS domains, respectively.

Pros and Cons: Static NS is a simple and easy-implemented negative sampling method. Compared with other heuristic sampling methods, Static NS has the characteristics of being fast, stable, and model-independent. The drawback of such methods is that the negative sampling distribution is fixed in the whole training process and the sampled negatives are not adaptive for each positive pair, resulting in sampling low-quality negatives. In general, performance increases as the number of negatives increases. However, too many negatives lead to an increase in GPU memory and training time. Therefore, the number of negative samples should be further investigated to accomplish the trade-off between performance and time consumption.

2) Hard Negative Sampling: Hard Negative Sampling (Hard NS) addresses the limitations of static NS by specifically targeting hard negatives. A concept of hard negative examples was first proposed in the bootstrapping method [105], [106], which treated incorrectly classified examples as hard examples. Over time, this idea evolved to not just include misclassified examples, but also samples that are close to the decision boundary in the feature space, irrespective of whether they have been previously misclassified or not. Thus, hard negatives are close to the decision boundary in the embedding space, which is difficult for a model to distinguish from positives.

Hard negative sampling dynamically selects hard negatives based on proximity in the feature space or based on the confidence level of the model's predictions. Based on the source of negatives, hard negative sampling can be divided into dynamic negative sampling (DNS) and mixture-based DNS. DNS dynamically mines hard negatives from the raw data while mixture-based DNS directly synthesizes negative embedding from the embedding space.

- *Dynamic Negative Sampling (DNS):* In DNS, negative samples are not predetermined but are selected based on the current state of the model or specific criteria that evolve as the training progresses. In light of the criteria, DNS can be divided into three groups, including anchor-based sampling, positive-based sampling, and anchor-positive sampling.

1) *Anchor-based DNS:* This group focuses on selecting negatives based on their relationship or similarity to a given anchor. The anchor acts as a reference, and negatives are chosen based on how they differ from this specific anchor in the feature space. The general distribution can be represented as $p_n(x^-) = \frac{S(f(x),f(x^-))}{\sum_{x' \in \mathcal{C}} S(f(x),f(x'))}$. Rendle et al. [61] developed an adaptive and efficient item sampler to select informative negative items with a small predicted rank, where the rank is designed as context-dependent and obtained by the score model among all items. Chen et al. [36] proposed an alternative sampler to replace word-popularity-based sampling in the skip-gram model, which prefers to select negative samples with higher inner product scores. Wu et.al [14] designed a distance-weighted sampling method to sample informative negatives where the distance weights between the anchor and negatives are clipped to correct the bias. In addition to inner product and distance, PageRank score can also be used to measure similarity in PinSage [39].
2) *Positive-based DNS:* Although anchor-based DNS offers better convergence and performance, it cannot take the impact of positive samples into consideration. It may seem intuitive that sampled negatives that are closer to positive samples will provide a sufficient risk of discrimination in order to distinguish positive from negative samples. The general distribution can be represented as $p_n(x^-) = \frac{S(f(x^+),f(x^-))}{\sum_{x' \in \mathcal{C}} S(f(x^+),f(x'))}$. Max Sampling [37] selected the most difficult negative answer by maximizing the similarities between the positive one and all negatives. Tran et al. [107] proposed a 2-stage sampling method, which first sampled a small subset of negative candidates and then selected informative negatives based on the similarity between a positive item and the candidates. Sun et al. [38] developed a ϵ-Truncated uniform negative sampling method to mine hard negative in KGE by using s-nearest neighbors of positive entities as candidates.
3) *Anchor-Positive DNS:* This method involves choosing negatives based on their joint relationship with both the anchor and the positive samples, which is valuable in complex learning scenarios like hinge loss frameworks. The general distribution can be represented as $p_n(x^-) = \frac{S(f(x),f(x^+),f(x^-))}{\sum_{x' \in \mathcal{C}} S(f(x),f(x^+),f(x'))}$. In computer vision, several works [41], [108], [109], [110] selected hard negative examples based on the triple loss, where negatives are close to the anchor and incur a high value of the loss. For WARP loss (Weighted Approximate-Rank Pairwise loss), a myriad of efforts [111], [112] adopted uniform negative sampling with rejection to mine hard negatives, which uniformly sampled a negative example until satisfied the constraint of $1 - S(f(x), f(x^+)) + S(f(x), f(x^-)) > 0$. Guo et al. [113] also designed a dynamic sampler in word embedding to select informative negative words with higher ranking scores than positive ones. Faghri et al. [114] combined anchor-based DNS and positive-based DNS to sample hard negative examples within each mini-batch in the image-caption retrieval domain. Guo et al. [115] selected high-ranked negative labels by a rank-invariant transformation for fast sampling. To improve the robustness, Several works [13], [116] proposed a semi-hard negative sampling method to select "semi-hard" negatives that were farther from the anchor-positive pair with a distance margin.

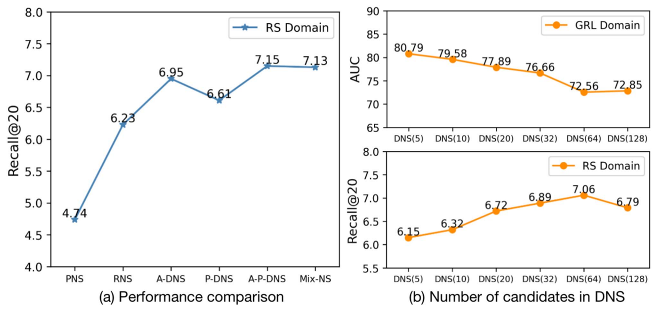


Fig. 5. (a) Performance comparison between Static NS and Hard NS. (b) Impact of the number of negative candidates in DNS.

- *Mixture-based DNS:* Inspired by Mixup [117], the idea of synthesizing hard negative in embedding space has become one of the popular negative sampling methods. Unlike methods that select samples existing in the dataset as negatives, Mixture-based DNS works at the level of embeddings, creating synthetic negative samples by strategically combining embeddings among several selected negatives. The synthetic negative samples can be represented as $\mathbf{x}^- = \text{Combine}(\{\mathbf{x}_i\})$, where $\mathbf{x}_i$ is a vector embedding of a selected sample x_i and $x_i \sim p_n(x^-)$. The function $\text{Combine}(\cdot)$ is utilized to combine the selected negatives for synthetic negative samples. In contrastive learning, MoChi [16] designed a hard negative mixing method to synthesize harder negatives by mixing the query and the hardest negatives in embedding space where the hardest ones are generated by anchor-based DNS. MixGCF [7] utilized the information propagation mechanism to design a hop-mixing strategy to synthesize negatives from multiple hops in a graph and adopted positive mixing to enhance negatives. MixKG [65] mixed candidate negative samples via the convex combination operation where candidate ones were filtered by two criteria: score-function based sampling and entity similarity correcting.

Pros and Cons: Hard NS is a model-dependent negative sampling technique that dynamically samples hard negative examples, which accelerates convergence and gives a clear gradient update direction for model training. Hard NS achieves significant performance improvements compared with static NS (e.g. PNS and RNS) (See Fig. 5(a)). However, such methods have an obvious disadvantage, namely that require more time to obtain distribution by the current model. To improve the efficiency, hard negative sampling usually samples a subset or takes mini-batch examples as negative candidates. Thus, the effectiveness of Hard NS relies on the number of candidates, which varies largely on different datasets (See Fig. 5(b)). More importantly, false negatives have gradually attracted more attention since they are closer to positive samples in embedding space and are difficult to distinguish from true negative samples. Many works [15], [86], [118], [119] focus on alleviating the false negative issue and boosting the robustness of the sampling process. Overall, Hard NS is an effective negative sampling method to speed up convergence and boost performance.

3) GAN-Based Negative Sampling: GAN-based negative sampling methods utilize generative adversarial networks (GANs) to mine or generate adversarial negatives, which are widely adopted in multiple domains. For GAN-based NS, the generator serves as a negative sampler to mine or generate informative negatives (false positives) to confuse the discriminator. The trainable models serve as a discriminator to distinguish the true positives and false positives (negative). Here, GAN-based NS methods can be roughly divided into two subcategories: GAN-based negative mining and GAN-based negative generation.

- *GAN-based Negative Mining:* GAN-based negative mining utilizes GAN to adaptively mine informative negatives where negatives are sampled from the discrete indexes of raw data. The general distribution can be represented as $p_n^G(x^-) = \frac{S(G(x),G(x^-))}{\sum_{x' \in \mathcal{C}} S(G(x),G(x'))}$ where G denotes a generator that generates negatives. SeqGAN [66] is the first work to utilize the generator to select discrete negatives and directly apply policy gradient to achieve gradient passing from the discriminator to the generator. IRGAN [43] utilized the generative information retrieval model to select the discrete index of samples as negatives, which uses REINFORCE [120] for model optimization. The abovementioned methods provide a new idea for negative sampling that designs an adversarial negative sampler and applies a policy gradient-based reinforcement learning (RL) method for model optimization. Such negative sampling methods are widely developed in other fields. For example, GraphGAN [44] employed a GAN-based framework for graph representation learning, and NMRN [45] proposed an adaptive negative sampler based on GAN in a streaming recommendation, and ACE [67] adopted GAN to find harder negatives in word embedding. In knowledge graph embedding, KBGAN [9] utilized one of the existing knowledge graph embedding models as the generator to sample high-quality negatives while Wang et al. [68] employed a two-layer fully-connected neural network as the generator for negative sampling.
- *GAN-based Negative Generation:* Instead of mining negatives from raw data, GAN-based negative generation aims to generate negative embedding for model optimization from embedding space. Such a method can generate synthetic samples which act as hard negatives. The general distribution can be represented as $p_n(\mathbf{x}^-) = \frac{S(\mathbf{x},\mathbf{x}^-)}{\sum_{x' \in \mathcal{C}} S(\mathbf{x},\mathbf{x}')}$, where $\mathbf{x}$, $\mathbf{x}^-$, and $\mathbf{x}'$ are generated vectors by the generator G. CFGAN [46] utilized the generator to generate continuous negative embeddings composed of real-valued elements in collaborative filtering without using the RL method for model optimization. DAML [75] employed adversarial learning to synthesize hard negatives from inputs (anchor, positive, and negative) by the generator for metric learning. Different from IRGAN comprising of two models, AdvIR [47] only used a single model to combine adversarial sampling and adversarial training to generate hard negative examples by adding adversarial perturbation vectors on negative example vectors. Similar to AdvIR, CLAE [73] designed an adversarial training algorithm for self-supervised learning that leveraged adversarial examples to generate harder negatives. HeGAN [70] extended GAN-based NS into Heterogeneous Information Networks (HIN), which generates "latent" negative nodes from a continuous distribution rather than true existing nodes. In dialogue systems, Gupta et al. [71] synthesized adversarial negative responses. AdCo [50] applied GAN-based NS to contrastive learning to generate challenging negative examples. Sinha et al. [72] proposed negative data augmentation (NDA) and integrated it into GAN where NDA acted as the generator to synthesize hard negatives for the discriminator. Chen et al. [121] utilized negative data augmentation (NDA) for novelty detection.

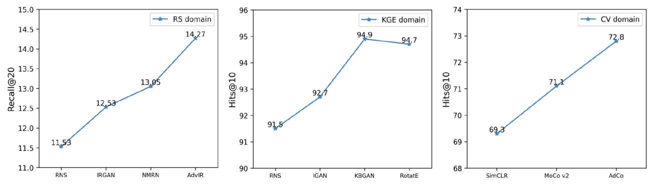


Fig. 6. Performance comparison between different GAN-based NS and RNS on recommendation system (left), knowledge graph embedding (middle), and computer vision (right).

Pros and Cons: GAN-based NS leverages adversarial learning to adaptively mine or generate negative samples, which provides more informative negatives and achieves faster convergence and better performance. This method brings about an adaptive negative sampler that acts as a generator for sufficiently exploiting training samples or embedding space to seek more informative negatives. As shown in Fig. 6 , compared to RNS, GAN-based NS achieves better performance on different domains since it mines more informative negatives for model training. However, GAN-based NS has an obvious shortcoming of unstable training. A common method to alleviate this issue is to pretrain the discriminator and the generator, which violates the original purpose of using negative sampling to speed up training. Thus, a stable optimization for GAN-based NS is a core and challenging problem, which leaves much room to investigate. Besides, GAN-based NS usually requires an additional network as the generator, which increases the training time and computational load. In general, GAN-based NS has a promising prospect for discovering hard negative samples but it is time-consuming, limiting its application to large-scale datasets.

4) Auxiliary-Based Negative Sampling: Auxiliary-based negative sampling methods rely on auxiliary information to sample negatives, comprising extra-data-based NS, graph-based NS, and cache-based NS. Each subcategory of Auxiliary-based NS possesses its unique characteristics, which will be demonstrated in detail.

- *Extra-data-based NS:* This method utilizes external datasets or additional information that is not part of the primary training set. The external data can provide a broader context or more examples, allowing for the selection of negatives that are more informative or challenging. For example, within the realm of recommendation systems, leveraging additional sources of information, such as social links, view data, and exposure data, can significantly enrich the negative sampling process. These auxiliary data types are particularly valuable as they offer insights into user preferences and behaviors that are not captured by traditional interaction data alone. SBPR [79] incorporated social links into negative sampling where social feedback served as negatives for positive items but was regarded as positives for unobserved negative items. Instead of utilizing a single additional information for negative sampling, PRFMC [80] leveraged both social links and geography to enhance negative sampling. MF-BPR [81] utilized multi-feedback data to compose negative items, which proposed a non-uniform negative sampler that quantified the impact of different types of feedback. Mix-Exp-NS [82] integrated view data into negative sampling, which designed a sampling weight to sample negatives from view and unobserved items. RNS-AS [83] combined exposure data and adversarial training into negative sampling to select high-quality real negatives. RecNS [84] mixed negatives sampled from positive-assisted and exposure-enhanced negative sampling methods to select hard and real negatives.

- *Graph-based NS:* This method capitalizes on the rich structural information inherent in graph data to enhance negative sampling. This approach is particularly pertinent in scenarios where data can be naturally represented as graphs, such as social networks, citation networks, or any domain where relationships between entities can be constructed as a graph. GNEG [11] leveraged random walk on the constructed word co-occurrence network to obtain negative sampling distribution. SamWalker [77] conducted the personalized random walk with a walking probability on a social network to obtain the distribution for sampling informative negative examples. SamWalker++ [122] designed a pseudo-social network to substitute the additional social network for random walk-based negative sampling. MCNS [6] approximated positive distribution with self-contrast approximation and utilized Metropolis-Hastings to accelerate negative sampling based on graph structure, which adopted DFS to traverse the graph for generating Markov chain. KGPolicy [40] integrated item knowledge graph into recommendation for negative sampling, which adopted reinforcement learning framework to select informative negative samples. MixGCF [7] utilized graph structure to sample hard negative items from multiple layers and mixed these with the positive item to obtain the synthetic negatives.
- *Cache-based NS:* This method employs a cache or memory bank as a dynamic repository for negative samples. In this way, the model can quickly and efficiently access these samples during training. This approach significantly reduces the computational overhead associated with sampling new negatives for each training iteration. Wu et al. [85] maintained a memory bank for storing negative samples. NSCaching [63] used a cache to maintain rare hard negative triplets and uniformly selected negatives from the cache for knowledge graph embedding learning. SRNS [86] designed a memory-based negative sampler to store high-variance negative candidates, which can effectively alleviate the false negative issue in recommendation. In unsupervised visual representation learning, MoCo [49] utilized a queue to store mini-batches data as negatives for the next iteration. MoCHi [16] and MoCoRing [18] utilize hard negative sampling methods to enhance Cache-based NS. GCC [88] maintained a queue to store negative instances for unsupervised graph representation learning. Cert [123] utilized a queue to store negative examples for language understanding. ESimCSE [12] and MoCoSE [64] leveraged a negative sample queue to further improve sentence embedding learning. Cache-based NS exploits more negatives during the training but requires more time to update the cache. Thus, a fast cache update mechanism should be explored to further improve the efficiency of cache-based NS.

Pros and Cons: Auxiliary-based NS leverages more auxiliary information to enhance the quality of negative samples. However, it's important to consider the trade-offs in terms of efficiency and domain-specific applicability. For example, the need to collect extra data can be a limitation, as it may not be feasible or practical in all domains. Additionally, the increased data volume can lead to higher data loading overheads, impacting computational efficiency. Graph-based NS depends on graph algorithms for traversing and mining negatives, which may increase the time consumption. Cache-based NS eliminates the constraint of GPU memory size for In-batch NS, but the update mechanism of the cache is very time-consuming. Thus, a trade-off between effectiveness and efficiency should be considered in negative sampling methods.

5) In-Batch Negative Sampling: In-batch negative sampling method is a distinct approach within the realm of negative sampling techniques, primarily relying on the composition of mini-batch data during training. Unlike other negative sampling methods that explicitly sample negatives for each positive pair, In-batch NS leverages the inherent structure of mini-batches to implicitly sample negatives. This method is particularly notable for its efficiency and simplicity. In-batch NS can be divided into three subcategories, including basic, debiased, and hard.

- *Basic In-batch NS:* This method is first proposed for neural network-based collaborative filtering [89] to utilize the non-linked examples within the same mini-batch as negatives, which boosts the training efficiency significantly. Ye et al. [124] leveraged In-batch NS to optimize the negative log-likelihood objective for unsupervised image embedding learning. In a dual-encoder model, In-batch NS has been used as a significant and effective trick for model training where $(B-1)$ examples in the same mini-batch are regarded as negatives [125], [126], [127], which efficiently boosts the number of negative training examples. In contrastive learning, SimCLR [48] leveraged In-batch

NS for visual representation learning where $2(B-1)$ examples generated by data augmentation are treated as negatives. Later, such a simple framework is widely used in various domains for unsupervised representation learning. SimCSE [97] follows SimCLR framework for sentence embedding learning. Several prior works [94], [95], [96], [128] on graph contrastive learning also adapt SimCLR framework to design augmentation transformations based on graph structure. SGL [91] incorporated self-supervised graph learning into recommendation and designed two types of In-batch NS for the auxiliary task.
- *Debiased In-batch NS:* While In-batch Negative Sampling stands as an efficient and broadly applicable method in various domains, it is not without its challenges. A primary concern is the issue of false negatives, which can significantly affect the performance of the model. In basic In-batch NS, samples within a mini-batch that are not explicitly labeled as positive are automatically treated as negatives. This assumption can lead to the inclusion of false negatives – actual positive samples mistakenly treated as negatives due to the absence of explicit positive labeling in the batch. DCL [15] proposed a debiased contrastive objective to alleviate sampling bias, which corrected the weights of negatives with positive examples in InfoNCE. DCLR [129] designed a debiased objective function that adjusted the weights of negatives using the complementary model. Huynh et al. [130] mitigated the effect of false negatives by two approaches: false negative elimination and attraction, which identified false negatives and removed them from the original negative candidates and added them into positive pairs set.
- *Hard In-batch NS:* Basic In-batch NS typically treats all samples within a mini-batch as negatives, assuming these samples are randomly sampled from the training data. However, this approach can overlook the importance of the difficulty level of negative samples. Hard In-batch NS method seeks to identify and utilize hard negatives within the batch. HCL [17] developed a hard negative sampling method for contrastive learning, which reweights the weights of negatives in the objective. Xiong et al. [104] globally selected hard negative samples by an asynchronously updated ANN index, in which negatives were generated by the dense retrieval (DR) model. Zhan et al. [131] adopted dynamic hard negatives to boost the training process and the ranking performance of the DR model. Yang et al. [52] proposed BatchSampler to globally sample hard negatives for contrastive learning.

Pros and Cons: In-batch NS stands out as an effective and efficient approach, particularly due to its ability to reuse samples from the current mini-batch without necessitating additional sampling operations. This method scales well with larger batch sizes, which facilitates the inclusion of more negative samples in the contrastive loss calculation, crucial for the training process. However, this means that the effectiveness of In-batch NS depends on the batch size. As indicated in Fig. 7 , the downstream performance exhibits a noticeable variation in relation to the batch size used during training. There is a clear trend indicating that larger batch sizes can improve downstream performance. In fact, In-batch NS is equivalent to random sampling since the data in the mini-batch are generated randomly. Compared to the basic In-batch NS leveraged by SimCLR, Debiased In-batch NS and Hard In-batch NS achieve better performance (See Fig. 7). Thus, integrating a hard negative mining strategy into In-batch NS is a huge challenge, which leaves us a lot of room for exploration. Furthermore, since In-batch NS is usually applied in contrastive learning for unsupervised representation learning, the false negative issue is an inevitable problem that plays a significant effect on the downstream performance.

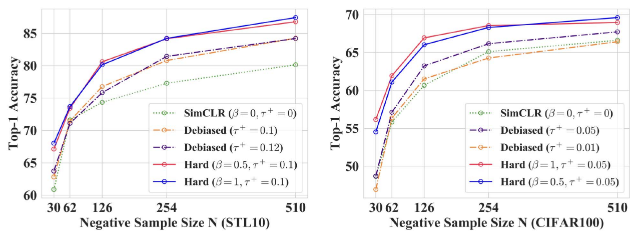


Fig. 7. Classification performance comparison with different In-batch NS on top-1 accuracy. Taken from [17].

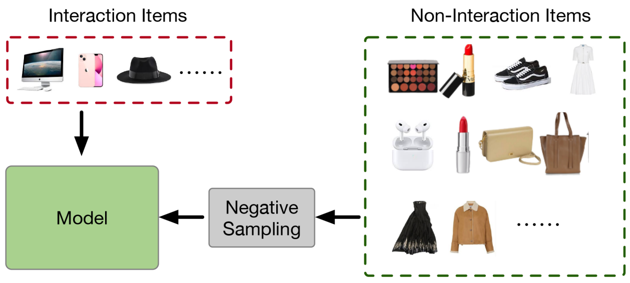


Fig. 8. Process of negative sampling in a recommender system where negative samples are sampled from non-interaction items.

IV. Applications

Negative sampling is an essential technique, which has been widely used in various domains(e.g. recommendation, graph representation learning, knowledge graph embedding, natural language processing, and computer vision).

A. Negative Sampling in RS

Recommender Systems (RS) have emerged as an indispensable tool for information filtering across a variety of online platforms, including e-commerce sites, advertising platforms, and entertainment services. The effectiveness of a recommender system hinges on its ability to model user preferences accurately, primarily based on historical interaction data. A significant challenge in RS is the inherent nature of interaction data, which typically consists of positive feedback (such as purchases, likes, or views) while lacking explicit negative feedback. To tackle this, negative sampling has been widely applied in previous works [2], [7], [29], [32].

As shown in Fig. 8, negative sampling aims to sample a small portion of items from the non-interaction items. The most popular negative sampling method is random negative sampling

(RNS) [2], [29], which serves as a basic negative sampling method to evaluate the performance of the proposed model. However, RNS assigns equal weights to unobserved items, which usually draws uninformative negative items for model training. To improve the quality of negative samples, many previous works mainly focus on popularity-based negative sampling (PNS) [132], [133], [134], hard negative sampling (hard NS) [19], [61], and GAN-based negative sampling (GAN-based NS) [45], [135]. The high-quality negative items contribute more to the gradient of the loss function, increasing its magnitude and accelerating convergence. PNS selects negative items based on item popularity, which depends on the data distribution. Intuitively, popular items that are non-interacted with a user are more likely to be negative. However, results in [54] reveal that the smoothing parameter for negative sampling has a significant effect on recommendation performances. Hard NS methods draw negative items based on prediction scores between the users and candidate negative items. Such methods aim to sample hard negative items with higher prediction scores. Additionally, GAN-based NS methods use the generator to obtain the sampling distribution for adaptively sampling informative negative items. However, the abovementioned methods suffer from the false negative issue where more hard (informative) negative items are more likely to be positive items. Synthesizing hard negative items has become an effective way to address this problem, which synthesizes or generates negatives in embedding space rather than sampling an existing item from raw data.

In recommender systems, the user-item interactions can be constructed as a bipartite graph where users/items are represented as nodes and observed interactions are regarded as edges [31], [32], [39], [136]. Even the user's historical behavior sequences can also be modeled as session graphs [137]. In light of this, graph learning methods provide a novel perspective for recommender systems, which incurs a myriad of GNN-based recommendation models in recent years. Similar to the traditional recommendation, negative sampling is also applied for model training in GNN-based recommendation models. Such sampling methods can incorporate graph information to sample negative items. For example, Pinsage [39] sampled hard negative items based on personalized PageRank scores for the Pinterest recommendation. MCNS [6] performed DFS on the graph to generate a Markov chain for negative sampling. KGPolicy [40] integrated a knowledge graph into negative sampling for seeking high-quality negative items. MixGCF [7] proposed hop mixing which mixed negative items sampled from different hops on the graph. To sum up, negative sampling in GNN-based recommendation can leverage graph information to design better negative sampling strategies but the efficiency issue should also be considered.

In summary, negative sampling has been proven to be an effective way to improve recommendation performance. As it applies to online services, recommendation systems must seriously consider the efficiency of negative sampling. In-batch NS is an efficient way that can be applied to large-scale recommendation. Thus, a challenging direction is how to reduce the sampling bias for In-batch NS.

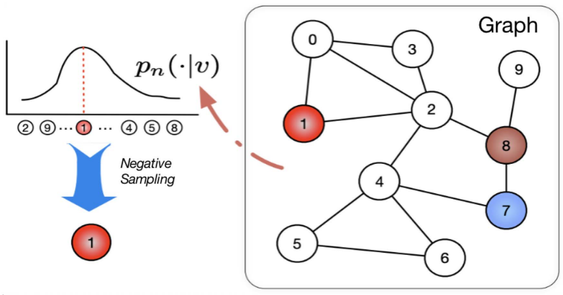


Fig. 9. Process of negative Sampling in GRL. Taken from [6].

B. Negative Sampling in GRL

Graph representation learning (GRL) has received a myriad of attention in recent years since graphs are a unique data structure that is employed extensively in many real-world applications, such as social networks, biological networks, academic networks, and many other domain-specific networks. The goal of graph representation learning is to learn low-dimensional embeddings for nodes that accurately reflect the structure and features of the original graph. After that, the learned embeddings can be used as feature inputs for downstream tasks, including node classification, link prediction, graph classification, and clustering.

Graph representation learning optimization involves computing the sum of all nodes in the graph, which is very time-consuming for large-scale graphs. To reduce the runtime of computation, negative sampling is designed to sample multiple negative nodes based on the noise distribution learned from the graph (See Fig. 9). DeepWalk [3] and Node2vec [5] followed the negative sampling setting in word2vec, which selected negative nodes according to the empirical unigram distribution proportional to the 3/4 power. However, this predefined sampling distribution can not dynamically sample negative nodes according to the training process. Gao et al. [138] proposed a self-paced negative sampling strategy to gradually sample the informative negative nodes for model optimization. Inspired by IRGAN, Gao et al. [138] incorporated generative adversarial network (GAN) into self-paced negative sampling to form an extension version of adversarial self-paced negative sampling. Robust-NS [139] argued that popularity-based negative sampling failed to accurately estimate the objective of skip-gram due to the popular neighbor problem, which proposed a distance-based negative sampler to draw negative nodes from candidate nodes without neighbors. As a most popular GNNs-based method, GraphSage [24] utilized negative sampling to optimize a graph-based loss function, which also kept the setting of word2vec. GraphGAN [44] integrated GAN into graph representation learning where negative samples were sampled by the generator. Later, MCNS [6] systematically analyzed the role of negative sampling in graph representation learning and proposed Markov chain Monte Carlo negative sampling.

Recently, contrastive learning has attracted a surge of interest for unsupervised visual representation learning [48], [49], [85]. In terms of the great success of contrastive learning in computer vision, numerous works extended it into graph learning [88],

[99]. After that, negative sampling in graph contrastive learning has also achieved tremendous success. Zhao et al. [99] proposed a graph-debiased contrastive learning framework to alleviate the false negative issue, which utilized clustering to obtain pseudo labels and conducted random sampling to sample negatives from the different clusters. Such a method suffers from expensive computational consumption. CuCo [100] proposed a curriculum contrastive learning framework that gradually samples negatives from easy to hard, in which hard negatives are determined by the score function. Zhu et al. [140] utilized heterogeneous graph structure to sample hard negatives with the largest similarities and synthesized more negatives by a mixing operation. ProGCL [101] utilized a beta mixture model (BMM) to mine true and hard negative examples for graph contrastive learning. Xiong et al. [104] proposed approximate nearest neighbor negative contrastive learning (ANCE) for the dense retrieval (DR) model, which globally sampled hard negatives with top retrieved scores from the current DR model.

In summary, negative sampling in graph representation learning has achieved some remarkable progress but still leaves a lot of room for improvement. For example, how to effectively use graph structure for negative sampling; how to incorporate GNNs propagation mechanism into negative sampling? How to design an effective negative sampling method for graph contrastive learning?

C. Negative Sampling in KGE

Knowledge graph embedding (KGE) provides a new way to represent knowledge by graphs, which constructs human knowledge as a knowledge graph consisting of entities, relationships, and semantic descriptions. A knowledge graph is a multi-relational graph and each relation can be represented as a triple of $< head\ entity, relation, tail\ entity >$. The key idea of knowledge graph embedding is to embed entities and relations in a KG into a continuous embedding space. The learned embeddings pave the way for many downstream applications, including knowledge graph completion, relation extraction, entity discovery, question answering, and recommender systems.

Negative sampling is a fundamental technique in knowledge graph embedding, which is applied to sample entities from the knowledge graph to replace the head entity or tail entity for forming a negative triple. A common negative sampling strategy in KGE is randomly sampling entities from a uniform distribution. However, such a strategy owns the obvious limitation that the sampled entities usually do not match the relations with the remaining entities, which does not provide meaningful information for gradient and prevents the model from learning better embeddings. To constrain the correlation of negative triples, [8] drew negative triples within the range constraints of entity types. PNS [141] proposed a probabilistic negative sampling to address the skewness issue in the dataset, which leveraged a tuning parameter to sample negatives from a pre-designed list that contained semantic possible negative instances. [142] developed two embedding-based negative sampling strategies: nearest neighbor sampling and near miss sampling, which aimed to search for negative triples that are close to positive triples in embedding space. KBGAN [9] developed an adversarial framework for knowledge graph embeddings, which utilized existing embedding models as the generator to generate high-quality negative triples. Later, several works [9], [68], [143] continued to integrate GAN into knowledge graph embedding for sampling negative triples. For example, IGAN [68] leveraged two-layer fully-connected neural networks as the generator to search informative negative triples. These methods sample high-quality negative triples with large gradients, which avoid the vanishing gradient issue and achieve performance improvements. However, GAN-based negative sampling methods need an extra generator and a complex gradient update approach. To efficiently sample negative triples, RotatE [143] proposed a self-adversarial negative sampling strategy that selected negatives based on the current model. Similar to DNS, Shan et al. [144] proposed a confidence-aware negative sampling method to enhance the confidence-aware knowledge representation learning (CKRL), which sampled negatives according to the softmax function that calculated by the current model on the candidate negatives set. SNS [145] utilized a distance-based score to search for high-quality negatives from a small randomly sampled candidate negative set in embedding space. NSCaching [63] used extra caches to store large-gradient negative heads and tails for each positive triplet respectively, and directly sampled negatives from the caches. To sample valid negative triples for assist model training, ANS [146] leveraged the K-Means clustering algorithm to measure the similarity in embedding space and uniformly sampled negatives from the same cluster for a particular positive triplet. SANS [76] utilized knowledge graph structure to sample negative triples where 1-hop neighbors are regarded as positives and the k-hop neighbors $(k > 1)$ serve as negatives (See Fig. 10). Instead of selecting an existing entity for negative sampling, MixKG [65] leveraged a mixing operation to synthesize hard negative samples. Different from the abovementioned methods of replacing head or tail entities, TransG [30] constructed negative triples by substituting the relation of triples. To sum up, negative sampling in a knowledge graph should focus on the validity of negatives satisfied the semantic relation, which can provide more information for the model to distinguish between positives and negatives.

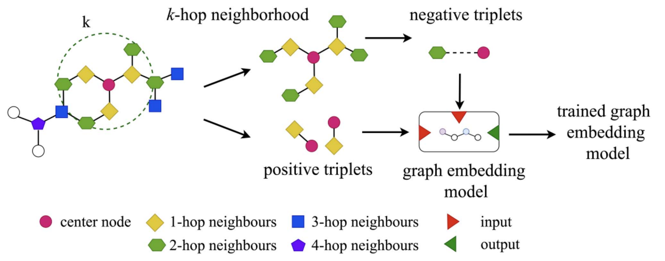


Fig. 10. Integrating the structure of knowledge graph into negative sampling. Taken from [76].

In summary, negative sampling in knowledge graph embedding relies heavily on the triple loss function. Thus, how to design a powerful loss function that incorporates the characteristics of a knowledge graph? How to develop a hierarchical negative sampling method by leveraging the hierarchical structure of the knowledge graph?

D. Negative Sampling in NLP

Negative sampling is a widely-used technique in natural language processing (NLP), which is widely used in word embedding, sentence embedding, dialogue systems, dense retrieval (DR), and named entity recognition (NER). Word2vec sampled negative words according to the distribution of word frequency. Goldberg et al. [147] attempted to explain negative sampling in word2vec. To alleviate the gradient vanishing issue in the skip-gram model with word-frequency negative sampling, Chen et al. [36] dynamically selected informative negative samples based on self-embedded features. Rao et al. [37] proposed three negative sampling strategies for negative answer selection, comprising random sampling, max sampling, and mix sampling. These sampling methods also be applied to open-domain dialogue systems for negative response selection [116]. Besides, Li et al. [116] also proposed a semi-hard sampling method where negatives satisfied a margin constraint. Negative sampling is an effective technique for named entity recognition (NER) models, which suffer greatly from unlabeled entity problems. Li et al. [148] leveraged negative sampling to randomly sample a small subset of unlabeled instances rather than the whole set. After that, Li et al. [149] designed a weighted sampling distribution to replace random sampling for boosting performance.

In sentence embedding learning [97], [150], [151], random sampling is also a common strategy for selecting negative samples. Guo et al. [150] proposed a hard negative mining method for bilingual sentence embedding learning, in which selected examples are close to the positive translation in semantic embedding space. Recently, contrastive learning for sentence embedding learning has achieved tremendous progress. For example, CLEAR [151] proposed sentence-level data augmentation for contrastive sentence representation learning, including word deletion, reordering, and substitution. SimCSE [97] proposed a model-level data augmentation that passed the same sentence twice with different dropout probabilities. The abovementioned methods focus on the sampling method for positive pairs where negatives come from the current mini-batch. CLINE [152] selected semantic negative examples in embedding space. MoCoSE [64] leveraged negative sample queue to obtain better performance. VaSCL [102] proposed neighborhood constrained contrastive learning, which utilized KNN to obtain top-K similar negatives from the current batch as hard negatives. SNCSE [103] utilized soft negative samples to enhance unsupervised sentence embedding learning where soft negatives were defined as the negation of original sentences with similar textual but different semantics. MixCSE [153] adopted a mixing operation to synthesize hard negatives for unsupervised sentence representation learning.

In recent years, dense retrieval (DR) models have become a dominant technique to solve the semantic match problem [126], [127], [154]. Negative sampling is an indispensable component for training DR models. EBR [155] adopted a random sampling strategy to select negatives for embedding-based retrieval models, which leveraged the triplet loss for recall optimization task. To improve the efficiency with the demand of a large number of negatives, several works [126], [127], [154] employed In-batch NS where other examples in the same mini-batch are treated as negatives. In fact, In-batch NS is approximately equivalent to random negative sampling. Moreover, the experimental results in RocketQA [127] demonstrated that it is beneficial to increase the number of negatives by introducing cross-batch negatives. Another popular direction is to apply hard negatives to train DR models. Gillick et al. [154] selected the most similar 10 entities based on the current model as hard negatives. Karpukhin et al. [126] utilized top passages generated by BM25 as hard negatives. Xiong et al. [104] proposed Approximate nearest neighbor Negative Contrastive Learning (ANCE) to select hard negatives by an ANN index. Different from static hard negative sampling methods, Zhan et al. [131] designed a dynamic hard negative sampling method, which utilized a trainable query encoder to retrieve top documents as hard negatives.

In summary, negative sampling in natural language processing still leaves a lot of room for exploration. For example, how to develop a better In-batch NS to mine hard negatives and simultaneously mitigate the false negative issue?

E. Negative Sampling in CV

After several decades of sustained effort, a large number of supervised methods have achieved significant improvements in computer vision (CV), which leveraged large datasets with labeled examples to learn visual representations. However, large-scale datasets with labels require human annotation, which is very expensive and hurts applications on the Internet scale. Contrastive learning for unsupervised visual representation learning becomes a natural way to address this issue, which aims to learn visual embeddings on unlabeled data. Recent developments [48], [49], [85], [156], [157] in unsupervised visual representation learning present a promising potential by using contrastive loss, which aims to contrastive positive pairs and negative pairs. Besides, metric learning is one of the basic learning ways for computer vision.

Generally, negative examples can be obtained either within a mini-batch or from a memory bank. In-batch negative sampling [48], [156] samples negatives from the current mini-batch. Memory-based negative sampling [85], [157], [158] samples negatives from a memory bank that stores mini-batch samples from previous batches. MoCo [49] argued that larger batch sizes play a significant role in model learning and maintained a queue to accumulate a large number of features that serve as negative samples for model training. SimCLR [48] proposed a simple framework for contrastive learning, which used all other images in the current batch as negative samples. Without the assistance of labels, the false negative issue is an inevitable problem in contrastive learning, which sampled true label examples from the data distribution. To address this, Chuang et al. [15] proposed a debiased contrastive loss (DCL) that corrected the weights of negatives in the objective. After that, Huynh et al. [130] proposed false negative cancellation strategies consisting of elimination and attraction to improve contrastive learning. Besides, Cai et al. [53] conducted an empirical study to analyze the importance of negative samples and concluded that only 5% hardest negatives are necessary for high-accuracy contrastive learning. Wang et al. [159] investigated the behavior of contrastive loss and concluded that loss optimization automatically focuses on hard negative samples which can be controlled by the temperature. To improve the performance and efficiency of contrastive learning, most strategies focus on hard negative sampling. MoCHi [16] proposed hard negative mixing strategies for contrastive learning to boost model learning, which synthesized hard negatives by mixing the hardest negatives and a positive query in the embedding space. Wu et al. [18] proposed conditional negative sampling to sample negatives within a ring where negatives in the ring are close but not too close to the positive example. Such a method is similar to the semi-hard negative sampling method. Similarly, Xie et al. [51] proposed four negative sampling strategies based on the cosine distance criterion between the anchor and negative candidates, comprised of hard, semi-hard, random, and semi-easy. HCL [17] proposed an efficient hard negative sampling method that rewrote the important weights of each negative example, which assigned higher weights to negatives that are close to the anchor. Different from hard negative sampling that only selects hard negatives, HCL adopted DCL to alleviate the false negative issue. Ge et al. [160] designed texture-based and patch-based negative sampling strategies to generate hard negative from the input images, which motivated the model to learn more semantics rather than superficial features. In addition, several works [50], [73] leveraged adversarial learning to improve contrastive learning. For example, CLAE [73] utilized adversarial examples and adversarial training for contrastive learning to generate hard negative examples. AdCo [50] adopted GAN for contrastive learning for sampling more informative negative samples. In addition, time-contrastive learning [161], [162], [163] is particularly applied to video data where positive and negative examples for learning are drawn from different timestamps within video sequences. In this framework, the same time step across different camera views is similar (positive samples), while frames from different time steps are dissimilar (negative samples).

Furthermore, numerous methods [13], [41], [110] in computer vision are designed on the triple loss based on metric learning by leveraging a max-margin approach to distinguish positive pairs from negative pairs, which are widely used in object detection, image classification, and face recognition. The easiest negative sampling method is random sampling. However, most randomly sampled negative examples are easy examples that easily satisfy the margin constraint, which contributes less to the gradients. To address this problem, [108] focused on mining hard negative examples that are close in embedding space to enhance the learning process. Wang et al. [110] mined hard negative triples after 10 epochs of training with randomly sampled negatives. However, mining hard negatives requires searching the whole training set, which is computationally expensive. Besides, the results in FaceNet [13] showed that the hardest negative examples significantly decrease the convergence speed and proposed a semi-hard negative mining strategy. To reduce the computational complexity, Mao et al. [164] sampled semi-hard negative examples from a mini-batch rather than the entire training set. Wu et al. [14] proposed a distance-weighted negative sampling strategy to optimize the triple loss. Iscen et al. [165] mined hard negatives for an anchor from its nearest euclidean

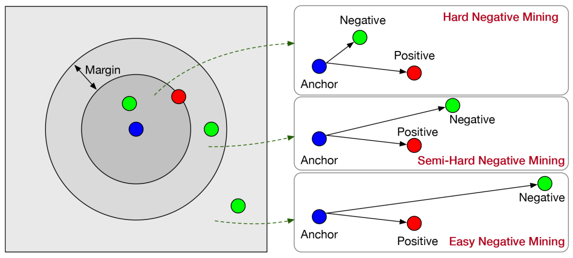


Fig. 11. Selection of negative examples in computer vision based on metric learning.

neighbors rather than manifold neighbors defined over the euclidean nearest neighbor graph. HDC [166] proposed a hard-aware deeply cascaded embedding to sample negatives from different hard levels of samples. Harwood et al. [59] proposed a smart negative sampling strategy for deep metric learning to sample a smart negative with a tuning variable and approximate positive examples. Although hard negatives provide large gradients for model optimization, are easy negatives really useless for metric learning? DAML [75] argued that easy negatives should not be neglected since these play a supplementary role in hard negatives, which utilized a hard negative generator to synthesize hard negatives from easy ones. Here, we plot a figure to intuitively illustrate the selection of negative examples in metric learning (See Fig. 11). In hard negative mining, the sampled hard negative examples are too close to the anchor, which leads to a high variance on the gradient and hinders the model from learning better representations. In semi-hard negative mining, the sampled negative examples fall in the margin region, which promises the similarity of the positive pair is higher than the negative one.

In summary, negative sampling in computer vision has left us a lot of room to explore. For example, how many negative samples are the best choices? Due to the emergence of negative-free methods, are negative samples really needed?

V. Discussion and Future Directions

In this section, we discuss several open problems in negative sampling and provide future directions for negative sampling to facilitate the development of this field. Here, we need to answer a series of questions. Is negative sampling necessary? If not, what kind of training paradigm is needed, and if so, how many negative samples are needed, and what quality of negative samples are needed?

Non-Sampling: Despite a myriad of negative sampling methods that have emerged recently, another hot direction is a non-sampling strategy that takes all negative samples into consideration in model optimization. Non-sampling strategy generally assigns lower weights for negative samples compared to positive ones. Such a setting is consistent with our intuition that positive samples should be evaluated with higher weights than negative ones. Due to the efficiency of non-sampling strategy, several efforts [167], [168], [169], [170] focus on promoting the efficiency of mini-batch Stochastic Gradient Descent (SGD) based methods. For example, Chen et al. [168] adopted non-sampling strategies for recommendation. Such a non-sampling strategy is also applicable for knowledge graph embedding learning [170] and word embedding [167]. Although the abovementioned works demonstrate that a non-sampling strategy can provide a more stable way for model optimization, its efficiency is a non-negligible issue. How to design a more efficient method for model optimization incorporated with a non-sampling strategy? Furthermore, a non-sampling strategy for other domains is not fully explored.

Getting Rid of Negative Sampling: Due to the quality and quantity of negative samples playing a significant impact on downstream performance, several works [171], [172], [173], [174] attempt to get rid of negative sampling and adopt other paradigms for model learning. SwAV [171] adopted online clustering to compare the consistency between cluster assignments rather than image features, which did not require explicit negative instances for unsupervised visual learning. BYOL [172] only utilized positive pairs without negative pairs for self-supervised learning, which adopted two neural networks to directly achieve prediction from one view to another view for the same image. SimSiam [173] presented a simple siamese network for representation learning, which also discards negative sample pairs. BGRL [174] also alleviates negative sampling for large-scale graph representation learning. Furthermore, generative self-supervised learning methods without negative sampling have been successfully applied in NLP [26], CV [175] and graph [176]. Thus, a promising research direction is to explore new learning methods or other alternatives to negative sampling.

The Quantity of Negative Samples: How many negative samples are needed? Arora et al. [177] proposed a theoretical analysis for contrastive learning to demonstrate a performance degradation by using larger negative samples. Wu et al. [178] proposed an adaptive negative sampling (ANS) method to dynamically adjust the ratio during the training process. Ash et al. [179] continued to investigate the number of negative examples in contrastive learning and revealed that the selection of optimal negative example size relies on the underlying concepts in the data. To address this gap between the theoretical analysis and empirical results, Nozawa et al. [180] proposed a lower bound for self-supervised learning, which aimed to adjust collision probability according to the number of negative samples. Recently, Awasthi et al. [181] argued that a collision-coverage trade-off is not an inherent property in contrastive learning, and claimed that the downstream performance does not degrade with the increasing of negative examples in a simple theoretical setting. Furthermore, Sohn [182] proposed an N-pair loss by adopting multiple negative samples for boosting deep metric learning. Although some works attempt to explain or explore the impact of negative sample size, there is still no one answer to what is the standard of negative sample size for a specific domain or even task. In practical applications, the quantity of negative samples is determined by a large number of trials. In the future, we are eager to obtain a criterion for a negative sample size.

The Quality of Negative Samples: Are the hardest negative samples the best ones? Does model optimization really not require easy negative samples? Results in several efforts demonstrated that easy negative samples play a crucial role in the early training stage. For example, Wang et al. [110] conducted random sampling to mine easy negatives in the first 10 epochs before mining hard negatives. CuCo [100] utilized curriculum learning to mine negative samples from easy to hard. DAML [75] highlighted the importance of easy negative samples for early stable training. Furthermore, many works have demonstrated that the hardest negative samples are not beneficial for model stability and robustness. For example, FaceNet [13] proposed semi-hard negative samples. Wu et al. [18] proposed to sample negatives with a ring constraint. Therefore, hard negative sampling should fully consider easy negatives and control the hardness of hard negatives. How to naturally and dynamically add easy negatives into hard negative sampling is a vital and worth exploring problem. Hardness of negative samples for model optimization is also a valuable research direction.

False Negative Issue: As a common and inevitable challenge for negative sampling, the false negative issue mainly comes from two aspects. The first is derived from unlabeled data in contrastive learning. Negative sampling in contrastive learning simultaneously focuses on mining hard negatives and mitigating the false negative issue. The second is that false negative instances naturally exist in implicit feedback in the recommendation domain. Although some strategies are employed to alleviate the false negative issue, it is impossible to eliminate this issue fundamentally under the paradigm of contrastive learning. Generative self-supervised learning can completely avoid this issue, which gets rid of negative sampling and becomes a powerful alternative for contrastive learning in unsupervised representation learning. In summary, alleviating or even eliminating the false negative issue is a meaningful research direction.

VI. Conclusion

In this survey, we have conducted an extensive review of the landscape of negative sampling techniques across a multitude of domains. Negative sampling is a fundamental technique in machine learning, which can accelerate the training process and boost downstream performance. We summarize a general negative sampling framework and develop a tool that contains many negative sampling methods among various domains. The selection methods for negative candidates are summarized, including global, local, mini-batch, hop, and memory-based. Besides, we categorize all existing negative sampling methods into five groups (static, hard, GAN-based, Auxiliary-based, and In-batch) and demonstrate their pros and cons. Furthermore, we illustrate negative sampling applications in various domains. Finally, open problems and future directions of negative sampling are presented.

References

[1] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[2] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," 2012, *arXiv:1205.2618*.

[3] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 701–710.

[4] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 1067–1077.

[5] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 855–864.

[6] Z. Yang, M. Ding, C. Zhou, H. Yang, J. Zhou, and J. Tang, "Understanding negative sampling in graph representation learning," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 1666–1676.

[7] T. Huang et al., "MixGCF: An improved training method for graph neural network-based recommender systems," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2021, pp. 665–674.

[8] D. Krompaß, S. Baier, and V. Tresp, "Type-constrained representation learning in knowledge graphs," in *Proc. Int. Semantic Web Conf.*, Springer, 2015, pp. 640–655.

[9] L. Cai and W. Y. Wang, "KBGAN: Adversarial learning for knowledge graph embeddings," 2017, *arXiv:1711.04071*.

[10] M. Grbovic, N. Djuric, V. Radosavljevic, F. Silvestri, and N. Bhamidipati, "Context-and content-aware embeddings for query rewriting in sponsored search," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2015, pp. 383–392.

[11] Z. Zhang and P. Zweigenbaum, "GNEG: Graph-based negative sampling for word2vec," in *Proc. Conf. Assoc. Comput. Linguistics*, 2018, pp. 566–571.

[12] X. Wu, C. Gao, L. Zang, J. Han, Z. Wang, and S. Hu, "ESimCSE: Enhanced sample building method for contrastive learning of unsupervised sentence embedding," 2021, *arXiv:2109.04380*.

[13] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A. unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 815–823.

[14] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krahenbuhl, "Sampling matters in deep embedding learning," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2840–2848.

[15] C.-Y. Chuang, J. Robinson, Y.-C. Lin, A. Torralba, and S. Jegelka, "Debiased contrastive learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 8765–8775.

[16] Y. Kalantidis, M. B. Sariyildiz, N. Pion, P. Weinzaepfel, and D. Larlus, "Hard negative mixing for contrastive learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 21798–21809.

[17] J. Robinson, C.-Y. Chuang, S. Sra, and S. Jegelka, "Contrastive learning with hard negative samples," 2020, *arXiv:2010.04592*.

[18] M. Wu, M. Mosse, C. Zhuang, D. Yamins, and N. Goodman, "Conditional negative sampling for contrastive learning of visual representations," 2020, *arXiv:2010.02037*.

[19] W. Zhang, T. Chen, J. Wang, and Y. Yu, "Optimizing top-n collaborative filtering via dynamic negative item sampling," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2013, pp. 785–788.

[20] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 2787–2795.

[21] T. Kipf, E. Van der Pol, and M. Welling, "Contrastive learning of structured world models," 2019, *arXiv:1911.12247*.

[22] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.

[23] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.

[24] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1025–1035.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[27] Y. Liu et al., "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.

[28] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proc. AAAI Conf. Artif. Intell.*, 2014, pp. 1112–1119.

[29] R. Pan et al., "One-class collaborative filtering," in *Proc. IEEE Int. Conf. Data Mining*, 2008, pp. 502–511.

[30] H. Xiao, M. Huang, Y. Hao, and X. Zhu, "TransG: A generative mixture model for knowledge graph embedding," 2015, *arXiv:1509.05488*.

[31] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 165–174.

[32] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and powering graph convolution network for recommendation," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 639–648.

[33] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 1. Ieee, 2005, pp. 886–893.

[34] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2009.

[35] T. Malisiewicz, A. Gupta, and A. A. Efros, "Ensemble of exemplar-SVMs for object detection and beyond," in *Proc. Int. Conf. Comput. Vis.*, 2011, pp. 89–96.

[36] L. Chen, F. Yuan, J. M. Jose, and W. Zhang, "Improving negative sampling for word representation using self-embedded features," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2018, pp. 99–107.

[37] J. Rao, H. He, and J. Lin, "Noise-contrastive estimation for answer selection with deep neural networks," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2016, pp. 1913–1916.

[38] Z. Sun, W. Hu, Q. Zhang, and Y. Qu, "Bootstrapping entity alignment with knowledge graph embedding.," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 4396–4402.

[39] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 974–983.

[40] X. Wang, Y. Xu, X. He, Y. Cao, M. Wang, and T.-S. Chua, "Reinforced negative sampling over knowledge graph for recommendation," in *Proc. Int. Conf. World Wide Web*, 2020, pp. 99–109.

[41] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 761–769.

[42] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[43] J. Wang et al., "IRGAN: A minimax game for unifying generative and discriminative information retrieval models," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2017, pp. 515–524.

[44] H. Wang et al., "Learning graph representation with generative adversarial nets," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 8, pp. 3090–3103, Aug. 2021.

[45] Q. Wang, H. Yin, Z. Hu, D. Lian, H. Wang, and Z. Huang, "Neural memory streaming recommender networks with adversarial training," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 2467–2475.

[46] D.-K. Chae, J.-S. Kang, S.-W. Kim, and J.-T. Lee, "CFGAN: A generic collaborative filtering framework based on generative adversarial networks," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2018, pp. 137–146.

[47] D. H. Park and Y. Chang, "Adversarial sampling and training for semi-supervised information retrieval," in *Proc. Int. Conf. World Wide Web*, 2019, pp. 1443–1453.

[48] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2020, pp. 1597–1607.

[49] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9729–9738.

[50] Q. Hu, X. Wang, W. Hu, and G.-J. Qi, "AdCo: Adversarial contrast for efficient learning of unsupervised representations from self-trained negative adversaries," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 1074–1083.

[51] J. Xie, X. Zhan, Z. Liu, Y. S. Ong, and C. C. Loy, "Delving into inter-image invariance for unsupervised visual representations," 2020, *arXiv:2008.11702*.

[52] Z. Yang et al., "BatchSampler: Sampling mini-batches for contrastive learning in vision, language, and graphs," 2023, *arXiv:2306.03355*.

[53] T. T. Cai, J. Frankle, D. J. Schwab, and A. S. Morcos, "Are all negatives created equal in contrastive instance discrimination?," 2020, *arXiv:2010.06682*.

[54] H. Caselles-Dupré, F. Lesaint, and J. Royo-Letelier, "Word2vec applied to recommendation: Hyperparameters matter," in *Proc. 12th ACM Conf. Recommender Syst.*, 2018, pp. 352–356.

[55] B. Yang, W.-T. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," 2014, *arXiv:1412.6575*.

[56] C. Tao, L. Mou, D. Zhao, and R. Yan, "RUBER: An unsupervised method for automatic evaluation of open-domain dialog systems," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 722–729.

[57] S. Ghazarian, J. T.-Z. Wei, A. Galstyan, and N. Peng, "Better automatic evaluation of open-domain dialogue systems with contextualized embeddings," 2019, *arXiv:1904.10635*.

[58] M. Bucher, S. Herbin, and F. Jurie, "Hard negative mining for metric learning based zero-shot classification," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2016, pp. 524–531.

[59] B. Harwood, V. KumarG, B. CarneiroReid, and T. Drummond, "Smart mining for deep metric learning," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2821–2829.

[60] D. Galanopoulos and V. Mezaris, "Hard-negatives or non-negatives? A hard-negative selection strategy for cross-modal retrieval using the improved marginal ranking loss," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 2312–2316.

[61] S. Rendle and C. Freudenthaler, "Improving pairwise learning for item recommendation from implicit feedback," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2014, pp. 273–282.

[62] X. Mao, W. Wang, Y. Wu, and M. Lan, "Boosting the speed of entity alignment 10×: Dual attention matching network with normalized hard sample mining," in *Proc. Int. Conf. World Wide Web*, 2021, pp. 821–832.

[63] Y. Zhang, Q. Yao, Y. Shao, and L. Chen, "NSCaching: Simple and efficient negative sampling for knowledge graph embedding," in *Proc. IEEE Int. Conf. Data Eng.*, 2019, pp. 614–625.

[64] R. Cao et al., "Exploring the impact of negative samples of contrastive learning: A case study of sentence embedding," 2022, *arXiv:2202.13093*.

[65] F. Che, G. Yang, P. Shao, D. Zhang, and J. Tao, "MixKG: Mixing for harder negative samples in knowledge graph," 2022, *arXiv:2202.09606*.

[66] L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence generative adversarial nets with policy gradient," in *Proc. AAAI Conf. Artif. Intell.*, 2017, pp. 2852–2858.

[67] A. J. Bose, H. Ling, and Y. Cao, "Adversarial contrastive estimation," 2018, *arXiv:1805.03642*.

[68] P. Wang, S. Li, and R. Pan, "Incorporating GAN for negative sampling in knowledge representation learning," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 2005–2012.

[69] H. Gao, J. Pei, and H. Huang, "ProGAN: Network embedding via proximity generative adversarial network," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 1308–1316.

[70] B. Hu, Y. Fang, and C. Shi, "Adversarial learning on heterogeneous information networks," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 120–129.

[71] P. Gupta, Y. Tsvetkov, and J. P. Bigham, "Synthesizing adversarial negative responses for robust response ranking and evaluation," 2021, *arXiv:2106.05894*.

[72] A. Sinha, K. Ayush, J. Song, B. Uzkent, H. Jin, and S. Ermon, "Negative data augmentation," 2021, *arXiv:2102.05113*.

[73] C.-H. Ho and N. Nvasconcelos, "Contrastive learning with adversarial examples," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 17081–17093.

[74] W. Wang, W. Zhou, J. Bao, D. Chen, and H. Li, "Instance-wise hard negative example generation for contrastive learning in unpaired image-to-image translation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 14020–14029.

[75] Y. Duan, W. Zheng, X. Lin, J. Lu, and J. Zhou, "Deep adversarial metric learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2780–2789.

[76] K. Ahrabian, A. Feizi, Y. Salehi, W. L. Hamilton, and A. J. Bose, "Structure aware negative sampling in knowledge graphs," 2020, *arXiv:2009.11355*.

[77] J. Chen, C. Wang, S. Zhou, Q. Shi, Y. Feng, and C. Chen, "SamWalker: Social recommendation with informative sampling strategy," in *Proc. Int. Conf. World Wide Web*, 2019, pp. 228–239.

[78] Y. Wang, Z. Liu, Z. Fan, L. Sun, and P. S. Yu, "DSKReG: Differentiable sampling on knowledge graph for recommendation with relational gnn," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2021, pp. 3513–3517.

[79] T. Zhao, J. McAuley, and I. King, "Leveraging social connections to improve personalized ranking for collaborative filtering," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2014, pp. 261–270.

[80] J. Manotumruksa, C. Macdonald, and I. Ounis, "A personalised ranking framework with multiple sampling criteria for venue recommendation," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2017, pp. 1469–1478.

[81] B. Loni, R. Pagano, M. Larson, and A. Hanjalic, "Bayesian personalized ranking with multi-channel user feedback," in *Proc. 10th ACM Conf. Recommender Syst.*, 2016, pp. 361–364.

[82] J. Ding, F. Feng, X. He, G. Yu, Y. Li, and D. Jin, "An improved sampler for Bayesian personalized ranking by leveraging view data," in *Proc. Int. Conf. World Wide Web*, 2018, pp. 13–14.

[83] J. Ding, Y. Quan, X. He, Y. Li, and D. Jin, "Reinforced negative sampling for recommendation with exposure data.," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 2230–2236.

[84] Z. Yang et al., "Region or global a principle for negative sampling in graph-based recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 6, pp. 6264–6277, Jun. 2023.

[85] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3733–3742.

[86] J. Ding, Y. Quan, Q. Yao, Y. Li, and D. Jin, "Simplify and robustify negative sampling for implicit collaborative filtering," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 1094–1105.

[87] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," 2020, *arXiv:2003.04297*.

[88] J. Qiu et al., "GCC: Graph contrastive coding for graph neural network pre-training," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 1150–1160.

[89] T. Chen, Y. Sun, Y. Shi, and L. Hong, "On sampling strategies for neural network-based collaborative filtering," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 767–776.

[90] K. Zhou et al., "S3-Rec: Self-supervised learning for sequential recommendation with mutual information maximization," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2020, pp. 1893–1902.

[91] J. Wu et al., "Self-supervised graph learning for recommendation," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 726–735.

[92] J. Yu, H. Yin, J. Li, Q. Wang, N. Q. V. Hung, and X. Zhang, "Self-supervised multi-channel hypergraph convolutional network for social recommendation," in *Proc. Int. Conf. World Wide Web*, 2021, pp. 413–424.

[93] X. Xia, H. Yin, J. Yu, Q. Wang, L. Cui, and X. Zhang, "Self-supervised hypergraph convolutional networks for session-based recommendation," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 4503–4511.

[94] K. Hassani and A. H. Khasahmadi, "Contrastive multi-view representation learning on graphs," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2020, pp. 4116–4126.

[95] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Deep graph contrastive representation learning," 2020, *arXiv:2006.04131*.

[96] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph contrastive learning with augmentations," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 5812–5823.

[97] T. Gao, X. Yao, and D. Chen, "Simcse: Simple contrastive learning of sentence embeddings," 2021, *arXiv:2104.08821*.

[98] L. Kong, C. d. M. d'Autume, W. Ling, L. Yu, Z. Dai, and D. Yogatama, "A mutual information maximization perspective of language representation learning," 2019, *arXiv:1910.08350*.

[99] H. Zhao, X. Yang, Z. Wang, E. Yang, and C. Deng, "Graph debiased contrastive learning with joint representation clustering.," in *Proc. Int. Joint Conf. Artif. Intell.*, 2021, pp. 3434–3440.

[100] G. Chu, X. Wang, C. Shi, and X. Jiang, "CuCo: Graph representation with curriculum contrastive learning.," in *Proc. Int. Joint Conf. Artif. Intell.*, 2021, pp. 2300–2306.

[101] J. Xia, L. Wu, G. Wang, J. Chen, and S. Z. Li, "ProGCL: Rethinking hard negative mining in graph contrastive learning," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2022, pp. 24332–24346.

[102] D. Zhang, W. Xiao, H. Zhu, X. Ma, and A. O. Arnold, "Virtual augmentation supported contrastive learning of sentence representations," 2021, *arXiv:2110.08552*.

[103] H. Wang, Y. Li, Z. Huang, Y. Dou, L. Kong, and J. Shao, "SNCSE: Contrastive learning for unsupervised sentence embedding with soft negative samples," 2022, *arXiv:2201.05979*.

[104] L. Xiong et al., "Approximate nearest neighbor negative contrastive learning for dense text retrieval," 2020, *arXiv:2007.00808*.

[105] H. Rowley, S. Baluja, and T. Kanade, "Human face detection in visual scenes," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 1995, pp. 875–881.

[106] K.-K. Sung and T. Poggio, "Example-based learning for view-based human face detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 1, pp. 39–51, Jan. 1998.

[107] V.-A. Tran, R. Hennequin, J. Royo-Letelier, and M. Moussallam, "Improving collaborative metric learning with efficient negative sampling," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 1201–1204.

[108] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer, "Discriminative learning of deep convolutional feature point descriptors," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 118–126.

[109] I. Loshchilov and F. Hutter, "Online batch selection for faster training of neural networks," 2015, *arXiv:1511.06343*.

[110] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 2794–2802.

[111] J. Weston, S. Bengio, and N. Usunier, "Wsabie: Scaling up to large vocabulary image annotation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2011, pp. 2764–2770.

[112] T. Zhao, J. McAuley, and I. King, "Improving latent factor models via personalized feature projection for one class recommendation," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2015, pp. 821–830.

[113] G. Guo, S. Ouyang, F. Yuan, and X. Wang, "Approximating word ranking and negative sampling for word embedding," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 4092–4098.

[114] F. Faghri, D. J. Fleet, J. R. Kiros, and S. Fidler, "VSE : Improving visual-semantic embeddings with hard negatives," 2017, *arXiv:1707.05612*.

[115] G. Guo, S. Zhai, F. Yuan, Y. Liu, and X. Wang, "VSE-ens: Visual-semantic embeddings with efficient negative sampling," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 290–297.

[116] J. Li, C. Tao, W. Wu, Y. Feng, D. Zhao, and R. Yan, "Sampling matters! an empirical study of negative sampling strategies for learning of matching models in retrieval-based dialogue systems," in *Proc. Conf. Empir. Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process.*, 2019, pp. 1291–1296.

[117] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," 2017, *arXiv:1710.09412*.

[118] T.-S. Chen, W.-C. Hung, H.-Y. Tseng, S.-Y. Chien, and M.-H. Yang, "Incremental false negative detection for contrastive learning," 2021, *arXiv:2106.03719*.

[119] C. Yang, Q. Wu, J. Jin, X. Gao, J. Pan, and G. Chen, "Trading hard negatives and true negatives: A debiased contrastive collaborative filtering approach," 2022, *arXiv:2204.11752*.

[120] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3, pp. 229–256, 1992.

[121] C. Chen et al., "Novelty detection via contrastive learning with negative data augmentation," 2021, *arXiv:2106.09958*.

[122] C. Wang, J. Chen, S. Zhou, Q. Shi, Y. Feng, and C. Chen, "SamWalker: Recommendation with informative sampling strategy," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 2, pp. 2004–2018, Feb. 2023.

[123] H. Fang, S. Wang, M. Zhou, J. Ding, and P. Xie, "CERT: Contrastive self-supervised learning for language understanding," 2020, *arXiv:2005.12766*.

[124] M. Ye, X. Zhang, P. C. Yuen, and S.-F. Chang, "Unsupervised embedding learning via invariant and spreading instance feature," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6210–6219.

[125] X. Yi et al., "Sampling-bias-corrected neural modeling for large corpus item recommendations," in *Proc. 13th ACM Conf. Recommender Syst.*, 2019, pp. 269–277.

[126] V. Karpukhin et al., "Dense passage retrieval for open-domain question answering," 2020, *arXiv:2004.04906*.

[127] Y. Qu et al., "RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering," 2020, *arXiv:2010.08191*.

[128] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Graph contrastive learning with adaptive augmentation," in *Proc. Int. Conf. World Wide Web*, 2021, pp. 2069–2080.

[129] K. Zhou, B. Zhang, W. X. Zhao, and J.-R. Wen, "Debiased contrastive learning of unsupervised sentence representations," 2022, *arXiv:2205.00656*.

[130] T. Huynh, S. Kornblith, M. R. Walter, M. Maire, and M. Khademi, "Boosting contrastive self-supervised learning with false negative cancellation," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2022, pp. 2785–2795.

[131] J. Zhan, J. Mao, Y. Liu, J. Guo, M. Zhang, and S. Ma, "Optimizing dense retrieval model training with hard negatives," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 1503–1512.

[132] O. Barkan and N. Koenigstein, "ITEM2VEC: Neural item embedding for collaborative filtering," in *Proc. IEEE 26th Int. Workshop Mach. Learn. Signal Process.*, 2016, pp. 1–6.

[133] H.-F. Yu, M. Bilenko, and C.-J. Lin, "Selection of negative samples for one-class matrix factorization," in *Proc. IEEE Int. Conf. Data Mining*, SIAM, 2017, pp. 363–371.

[134] D. Lian, Q. Liu, and E. Chen, "Personalized ranking with importance sampling," in *Proc. Web Conf.*, 2020, pp. 1093–1103.

[135] B. Jin et al., "Sampling-decomposable generative adversarial recommender," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 22629–22639.

[136] R. v. d. Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," 2017, *arXiv:1706.02263*.

[137] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 346–353.

[138] H. Gao and H. Huang, "Self-paced network embedding," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 1406–1415.

[139] M. Armandpour, P. Ding, J. Huang, and X. Hu, "Robust negative sampling for network embedding," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 3191–3198.

[140] Y. Zhu, Y. Xu, H. Cui, C. Yang, Q. Liu, and S. Wu, "Structure-enhanced heterogeneous graph contrastive learning," in *Proc. SIAM Int. Conf. Data Mining*, SIAM, 2022, pp. 82–90.

[141] V. Kanojia, H. Maeda, R. Togashi, and S. Fujita, "Enhancing knowledge graph embedding with probabilistic negative sampling," in *Proc. 26th Int. Conf. World Wide Web Companion*, 2017, pp. 801–802.

[142] B. Kotnis and V. Nastase, "Analysis of the impact of negative sampling on link prediction in knowledge graphs," 2017, *arXiv:1708.06816*.

[143] Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang, "RotatE: Knowledge graph embedding by relational rotation in complex space," 2019, *arXiv:1902.10197*.

[144] Y. Shan, C. Bu, X. Liu, S. Ji, and L. Li, "Confidence-aware negative sampling method for noisy knowledge graph embedding," in *Proc. IEEE Int. Conf. Big Knowl.*, 2018, pp. 33–40.

[145] M. K. Islam, S. Aridhi, and M. Sma ïl-Tabbone, "Simple negative sampling for link prediction in knowledge graphs," in *Proc. Int. Conf. Complex Netw. Appl.*, Springer, 2021, pp. 549–562.

[146] S. Qin, G. Rao, C. Bin, L. Chang, T. Gu, and W. Xuan, "Knowledge graph embedding based on adaptive negative sampling," in *Proc. Int. Conf. Pioneering Comput. Scientists, Engineers Educators*, Springer, 2019, pp. 551–563.

[147] Y. Goldberg and O. Levy, "word2vec explained: Deriving Mikolov et al.'s negative-sampling word-embedding method," 2014, *arXiv:1402.3722*.

[148] Y. Li, L. Liu, and S. Shi, "Empirical analysis of unlabeled entity problem in named entity recognition," 2020, *arXiv:2012.05426*.

[149] Y. Li, L. Liu, and S. Shi, "Rethinking negative sampling for handling missing entity annotations," in *Proc. Conf. Assoc. Comput. Linguistics*, 2022, pp. 7188–7197.

[150] M. Guo et al., "Effective parallel corpus mining using bilingual sentence embeddings," 2018, *arXiv:1807.11906*.

[151] Z. Wu, S. Wang, J. Gu, M. Khabsa, F. Sun, and H. Ma, "CLEAR: Contrastive learning for sentence representation," 2020, *arXiv:2012.15466*.

[152] D. Wang, N. Ding, P. Li, and H.-T. Zheng, "CLINE: Contrastive learning with semantic negative examples for natural language understanding," 2021, *arXiv:2107.00440*.

[153] Y. Zhang, R. Zhang, S. Mensah, X. Liu, and Y. Mao, "Unsupervised sentence representation via contrastive learning with mixing negatives," in *Proc. AAAI Conf. Artif. Intell*, 2022, pp. 11730–11738.

[154] D. Gillick et al., "Learning dense representations for entity retrieval," 2019, *arXiv:1909.10506*.

[155] J.-T. Huang et al., "Embedding-based retrieval in Facebook search," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 2553–2561.

[156] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018, *arXiv:1807.03748*.

[157] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2020, pp. 776–794.

[158] I. Misra and L. v. d. Maaten, "Self-supervised learning of pretext-invariant representations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 6707–6717.

[159] F. Wang and H. Liu, "Understanding the behaviour of contrastive loss," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 2495–2504.

[160] S. Ge, S. Mishra, C.-L. Li, H. Wang, and D. Jacobs, "Robust contrastive learning using negative samples with diminished semantics," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 27356–27368.

[161] P. Sermanet, K. Xu, and S. Levine, "Unsupervised perceptual rewards for imitation learning," 2016, *arXiv:1612.06699*.
[162] P. Sermanet et al., "Time-contrastive networks: Self-supervised learning from video," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 1134–1141.
[163] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3M: A universal visual representation for robot manipulation," 2022, *arXiv:2203.12601*.
[164] C. Mao, Z. Zhong, J. Yang, C. Vondrick, and B. Ray, "Metric learning for adversarial robustness," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 478–489.
[165] A. Iscen, G. Tolias, Y. Avrithis, and O. Chum, "Mining on manifolds: Metric learning without labels," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7642–7651.
[166] Y. Yuan, K. Yang, and C. Zhang, "Hard-aware deeply cascaded embedding," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 814–823.
[167] X. Xin, Y. Fajie, H. Xiangnan, and J. Joemon, "Batch is not heavy: Learning word embeddings from all samples," in *Proc. Conf. Assoc. Comput. Linguistics*, 2018, pp. 107–132.
[168] C. Chen et al., "An efficient adaptive transfer neural network for social-aware recommendation," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 225–234.
[169] C. Chen, M. Zhang, Y. Zhang, Y. Liu, and S. Ma, "Efficient neural matrix factorization without sampling for recommendation," *ACM Trans. Inf. Syst.*, vol. 38, no. 2, pp. 1–28, 2020.
[170] C. Chen, M. Zhang, W. Ma, Y. Liu, and S. Ma, "Jointly non-sampling learning for knowledge graph enhanced recommendation," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 189–198.
[171] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, vol. 33, pp. 9912–9924, 2020.
[172] J.-B. Grill et al., "Bootstrap your own latent-a new approach to self-supervised learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 21271–21284.
[173] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 15750–15758.
[174] S. Thakoor et al., "Large-scale representation learning on graphs via bootstrapping," 2021, *arXiv:2102.06514*.
[175] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 16000–16009.
[176] Z. Hou et al., "GraphMAE: Self-supervised masked graph autoencoders," 2022, *arXiv:2205.10803*.
[177] S. Arora, H. Khandeparkar, M. Khodak, O. Plevrakis, and N. Saunshi, "A theoretical analysis of contrastive unsupervised representation learning," 2019, *arXiv:1902.09229*.
[178] C. Wu, F. Wu, and Y. Huang, "Rethinking infonce: How many negative samples do you need?," 2021, *arXiv:2105.13003*.
[179] J. T. Ash, S. Goel, A. Krishnamurthy, and D. Misra, "Investigating the role of negatives in contrastive representation learning," 2021, *arXiv:2106.09943*.
[180] K. Nozawa and I. Sato, "Understanding negative samples in instance discriminative self-supervised representation learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 5784–5797.
[181] P. Awasthi, N. Dikkala, and P. Kamath, "Do more negative samples necessarily hurt in contrastive learning?," 2022, *arXiv:2205.01789*.
[182] K. Sohn, "Improved deep metric learning with multi-class N-pair loss objective," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 1849–1857.

Zhen Yang received the master's degree from the Institute of Microelectronics, Tsinghua University. She is currently working toward the PhD degree with the Department of Computer Science and Technology, Tsinghua University. Her research interests include graph representation learning, graph-based recommendation, contrastive learning, and large language models (LLMs). She has published some papers on top conferences and journals, such as KDD, WWW, AAAI, ICCV and *IEEE Transactions on Knowledge and Data Engineering*.

Ming Ding received the PhD degree from the Department of Computer Science and Technology, Tsinghua University, in 2023. He is currently working with Zhipu AI. His research interests include graph learning, natural language processing and cognitive artificial intelligence. He has published many papers on top conferences, such as NeurIPS, KDD, ACL, IJCAI, etc.

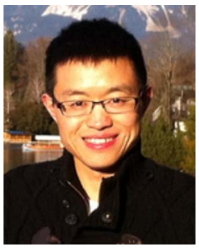
Tinglin Huang received the master's degree from Zhejiang University. He is currently working toward the PhD degree with the Department of Computer Science, Yale University. His research interests include computational biology, geometric deep learning, and graph learning. He has published some papers at top conferences, such as NeurIPS, KDD, WWW, etc.

Yukuo Cen received the bachelor's degree from Tsinghua University, and the PhD degree from the Department of Computer Science and Technology, Tsinghua University, in 2023. He is currently working with Zhipu AI. He has published several papers on the top international conferences and journals, including KDD, WWW, and *IEEE Transactions on Knowledge and Data Engineering*, on graph representation learning and recommender systems.

Junshuai Song received the PhD degree from Peking University, in 2021. He has published papers in top conferences and journals, including *IEEE Transactions on Knowledge and Data Engineering*, ICDE, ACL, and AAAI on graph mining, network representation learning, and recommender systems.

Bin Xu (Member, IEEE) is a professor in computer science with Tsinghua University. He became ACM professional member, in 2009. His research interest is knowledge gragh and large language models (LLMs). He is core member to develop GLM family of LLMs.

Yuxiao Dong is an associate professor in computer science with Tsinghua University. His research focuses on data mining, graph machine learning, and foundation models. Together with collaborators, his recent research includes GNNs (hetero. graph transformer, GRAND, OGB), network embedding (metapath2vec, NetMF, NetSMF, SketchNE), graph pretraining (GraphMAE, GPT-GNN, GCC), and LLMs (GLM-130B, ChatGLM, CodeGeeX). He received the 2022 SIGKDD Rising Star Award.


Jie Tang (Fellow, IEEE) is a WeBank chair professor in computer science with Tsinghua University. He is a fellow of the ACM, a fellow of AAAI. His research interest is in artificial general intelligence (AGI). His research has received the SIGKDD Test-of-Time Award (10-year Best Paper). He also received the SIGKDD Service Award. Recently, he puts all his efforts into Large Language Models (LLMs). He invented AMiner.org and developed the GLM family of LLMs, such as ChatGLM, CogVLM, CodeGeeX, and CogView. He served as general chair of WWW'23, program chair of WWW'22 and WSDM'15, and EiC of IEEE Trans. on Big Data and AI Open J.