

Unleashing the Power of Emojis in Texts via Self-supervised Graph Pre-Training

Zhou Zhang^{1*}, Dongzeng Tan^{2*}, Jiaan Wang¹, Yilong Chen¹, Jiarong Xu^{1†}



¹Fudan University, ²Alibaba Group

Correspondence: zhouzhang23@m.fudan.edu.cn, erictandz@gmail.com, jawang.nlp@gmail.com, yilongchen@uchicago.edu, jiarongxu@fudan.edu.cn

Abstract

Emojis have gained immense popularity on social platforms, serving as a common means to supplement or replace text. However, existing data mining approaches generally either completely ignore or simply treat emojis as ordinary Unicode characters, which may limit the model’s ability to grasp the rich semantic information in emojis and the interaction between emojis and texts. Thus, it is necessary to release the power of emojis in social media data mining. To this end, we first construct a heterogeneous graph consisting of three types of nodes, *i.e.*, post, word and emoji nodes to improve the representation of different elements in posts. The edges are also well-defined to model how these three elements interact with each other. To facilitate the sharing of information among post, word and emoji nodes, we propose a graph pre-training framework for text and emoji co-modeling, which contains two graph pre-training tasks: node-level graph contrastive learning and edge-level link reconstruction learning. Extensive experiments on the Xiaohongshu and Twitter datasets with two types of downstream tasks demonstrate that our approach proves significant improvement over previous strong baseline methods.¹

1 Introduction

Emojis have gained significant attention due to their popularity in digital communication, especially on social media platforms. They convey emotions and reactions, supplementing or replacing text in posts, which adds nuance to conversations. For example, the heart emoji  conveys love more strongly than words, and the face-with-tears-of-joy emoji  effectively expresses humor. Accurately modeling emojis can enhance natural language processing

(NLP) tasks like sentiment analysis and emoji generation.

Current methods for processing emojis are varied. Traditional statistical methods either ignore emojis or treat them as regular characters, failing to capture their semantics. Methods like emoji2vec (Eisner et al., 2016) learn emoji representations based on textual descriptions but rely heavily on annotations, which fails to keep up with the evolving semantics of emojis in different contexts. Self-supervised learning, which doesn’t require labeled data, is widely used for emojis. However, many self-supervised methods either use emojis only as annotations (Felbo et al., 2017) for text embeddings or focus solely on emoji embeddings, neglecting text embeddings. These models fail to fully utilize the relationship between text and emojis.

To address these issues, we aim to better model the relationship between emojis and text, improving embeddings by learning their interactions. It can be observed that emojis in similar texts share semantics, and texts with similar emojis often have shared sentiments or popularity.

Given the ability of heterogeneous graphs to model interactive relations across information at different granularities, we construct a heterogeneous graph to represent the information that lies in social media posts. Specifically, the graph contains emoji nodes and word nodes to capture the relationship between emojis and texts. To better absorb the global information of the whole post, another node type, *i.e.*, post nodes, is also constructed in our graph. In this manner, the above similarity could be represented by the similarity between n -hop neighbors.

In order to make information further flow between different nodes in the heterogeneous graph, we develop two self-supervised graph pre-training tasks that facilitate reciprocal text-emoji joint representation learning. (1) Based on the observa-

*Both authors contributed equally to this research.

†Corresponding author.

¹<https://github.com/ginkoeric/Self-supervised-Graph-Pre-training-for-Emoji>

tions above, we construct *node-level subgraph contrastive learning task*. This task first randomly chooses a starting node, and then samples its neighbors through random walk, subsequently forming a sub-graph containing both the starting node and the neighbors. By repeating this process on different starting nodes, several sub-graphs are obtained. The task is used to guide the graph model to judge whether two sub-graphs are generated from the same starting node. This task could help capture the similarity among nodes. (2) To better capture the mutual relationships among these three types of nodes, we build the second task: *edge-level link reconstruction learning*. This task selects existing edges as positive edges and constructs negative edges from randomly added non-existing ones. Then, the task requires the graph model to distinguish positive edges from negative ones. This task lets the graph model focus on the relationships among different node types (every edge links two nodes from different types). Consequently, with the help of the above two tasks, graph models could learn to understand how posts, emojis and words interact with each other, which will benefit downstream tasks like popularity prediction.

Our main contributions are concluded as follows:

- To better describe the relationship between posts, emojis and words in social media data, we construct a post-emoji-word heterogeneous graph.
- We design a graph pre-training framework to learn the post, emoji, and word-level embedding. This framework consists of two self-supervised tasks on both nodes and edges in the heterogeneous graph. The embedding learned has strong transferability, which could be integrated with existing language models' text/token embedding, and improve their emoji understanding ability, and then be used in various downstream tasks.
- We conduct extensive experiments on datasets from different social media platforms (*i.e.*, Xiaohongshu and Twitter) to demonstrate that our approach achieves better performance on various downstream tasks (*e.g.* popularity prediction and sentiment prediction) than strong baselines. Our model outperforms the baseline model by 2% - 10%.

2 Related Work

Emojis play multiple roles in people's communication. They not only provide an expressive way to convey emotions or situational information but are

also convenient for grasping implicit sentiments as well as emotions. The use of emojis on social media tends to affect the popularity of tweets. However, the importance of emojis in textual studies has not been fully realized by early work (Hotho et al., 2005; Mikolov et al., 2013; Allahyari et al., 2017). Recently, many studies have revealed the significance of emojis, and these studies could be summarized into four types of research fields: traditional pre-process, description mining, supervised learning, and self-supervised learning methods.

Traditional Pre-process Methods. These methods directly utilize standard pre-processing methods and then feed the pre-processed emojis to language models (Wijeratne et al., 2017; Pavan Kumar and Dhinesh Babu, 2019; Yinxia et al., 2020). Specifically, (1) *Removing emoji strategy* treats emojis as noise and remove them from the original text (Pavan Kumar and Dhinesh Babu, 2019). This strategy neglects the semantic information conveyed by emojis. (2) *Keeping emoji strategy* retains emojis as tokens, but unfortunately, the tokens fail to convey the accurate meanings carried by the emojis (Yinxia et al., 2020). (3) *Translating emoji* translates emojis into descriptive texts (Wijeratne et al., 2017) and learn representations from these descriptions. Though these methods are intuitive, to perform pre-processing on emojis, we have to maintain a dictionary that covers specific types of emojis (*e.g.*, iOS emojis encoded by Unicode). Such a dictionary cannot be generalized to other types of emojis (*e.g.*, Xiaohongshu emojis and user-created emojis).

Description Mining Methods. To further capture the emojis' semantics, another line of research investigates methods for learning emoji representations by utilizing the corresponding textual descriptions (Eisner et al., 2016). Yinxia et al. (2020) introduce an attention model to learn representations for emojis and text, while simultaneously conducting sentiment classification. This approach learns emoji representations by minimizing the similarity between the representation of an emoji and its corresponding textual description. However, the effectiveness of this approach relies on the availability of textual descriptions for emojis, which may not be accessible for certain types of emojis, such as user-created emojis.

Supervised Learning Methods. The third line of work explores the use of emojis to assist in various downstream tasks, such as sentiment classification, emotion analysis, and emoji prediction (Chen

et al., 2019b; Singh et al., 2022; Zhao et al., 2023). Researchers create annotated datasets where text inputs are paired with corresponding emojis, enabling models to discern the connection between textual contexts and emojis implicitly. For instance, Chen et al. (2019b) tackle the challenge of cross-lingual sentiment classification by using emojis as an instrument because similar emojis convey similar emotions across different languages. These methods are limited to only leveraging the emoji meaning related to certain tasks at hand, e.g., emojis employed for sentiment classification may not effectively represent the semantic and concrete meaning of an emoji like “🚗”.

Self-Supervised Learning Methods. The fourth line of existing studies focuses on utilizing emoji information and learning emoji representations via self-supervised learning. In detail, these methods leverage self-supervised (or unsupervised) tasks to incorporate emoji information into text representations and enhance the meaningfulness of emojis. For instance, Felbo et al. (2017); Park et al. (2018) propose DeepMoji which involves identifying whether a post contains specific emojis. Chen et al. (2019a) further develop SEntiMoji for sentiment analysis based on DeepMoji. These approaches allow for the integration of emoji information into text representations. Another study by Illendula and Yedulla (2018) constructs an emoji co-occurrence network and learns emoji representations by minimizing the similarity between neighboring emojis. Nevertheless, these methods have a limitation as they do not consider the positional relationship between the text and emojis. Also, emojis are used only as annotations in these methods and they are trained solely based on specific tasks. The lack of universal emoji embedding will make it difficult for these methods to be applied to a wide range of downstream tasks. To bridge this gap, our study investigates how to incorporate the positional information and utilize the relation between text and emojis to learn universal embeddings of emojis.

Different from previous studies, in this work, a heterogeneous graph is constructed to depict the connections among posts, words, and emojis. Besides, two self-supervised learning tasks are designed: node-level subgraph contrastive learning and edge-level link reconstruction learning, which facilitate the simultaneous refinement of posts, words, and emoji representations.

3 Methodology

In this section, we first construct the heterogeneous graph with three types of nodes: posts, emojis and words, to capture the information and interactions among them (§ 3.1). Then, based on the heterogeneous graph, we design a graph pre-training framework to model the comprehensive interaction among all types of nodes (§ 3.2). After that, we show how to use our proposed method in various downstream tasks, *i.e.*, popularity prediction and sentiment classification (§ 3.3). Figure 1 illustrates the overview of our pre-training and fine-tuning stages.

3.1 Heterogeneous Graph Construction

To incorporate the relationship and information between textual content and emojis, we build up a heterogeneous graph, as shown in Figure 1(a), which consists of three types of nodes:

- **Post node:** $\mathcal{T} = \{t_1, t_2, \dots, t_N\}$ represents the set of post nodes, where $t_i (i = 1, 2, \dots, N)$ is the i -th post node containing only the textual information in the i -th post, and N is the total number of posts.
- **Word node:** $\mathcal{W} = \{w_1, w_2, \dots, w_M\}$ is the set of word nodes (*i.e.*, the word vocabulary), where M is the vocabulary size. \mathcal{W} includes all the words collected from all the posts.
- **Emoji node:** $\mathcal{E} = \{e_1, e_2, \dots, e_K\}$ means the set of emoji nodes, comprising the top K emojis with the highest occurrences in posts.

Among these nodes, there exist three kinds of edges to describe their connections.

- **Post-emoji edge:** This type of edge indicates that if an emoji is in/out of the post. In this manner, we can build connections among posts that share the same emojis, and lead to the information flow from post to emoji. For example, posts “I didn’t pass the exam 😞” and “I finally pass the exam 😄” tell totally different emotions with the same emoji 😞. The former is a sad feeling while the latter means an overwhelming joyful feeling opposite. Therefore, connections between posts and emojis will help a lot in an informative representation study for both.
- **Post-word edge:** This edge type shows the existence of a word in the post. With no doubt that building connections between all words from the post and itself will make the graph complex and may introduce uncontrollable noise. To avoid that, we treat the words inside hashtag as key-

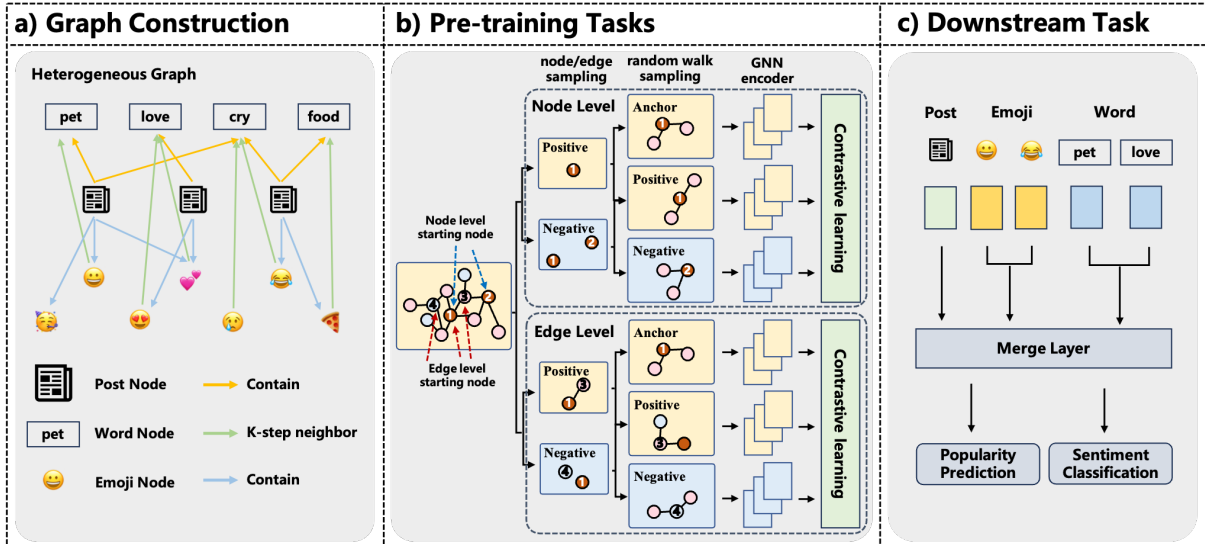


Figure 1: An illustration of all processes of our model. Part a) shows the construction of our heterogeneous graph that incorporates post, emoji and word nodes, along with their connections. Part b) shows the structure of node level and edge level pre-training tasks. Part c) shows the structure of two downstream tasks.

words, and then filter keywords from posts and only consider links between them. Typically, the hashtag of a post matches its topic. In this way, we can make sure that connected posts will share the same topic, and make the textual semantic information more finely.

- **Word-emoji edge:** This type of edge comes from the emoji and its k -step neighbor words. For instance, in the sentence “What a nice 🍷 day!”, words “a”, “nice”, and “day” are 🍷’s two-step neighbors. They contribute to a fine-grained semantic knowledge for emojis, and help us to locate emojis’ position in a post.

To initialize the node embeddings, we use the pre-trained language model (i.e., BERT (Devlin et al., 2018)) to generate representations for post nodes and word nodes. For emoji nodes, we obtain the initialization by randomization, these will be discussed comprehensively in Section 4.

3.2 Graph Pre-training Framework

Based on the heterogeneous graph built above, we design a graph pre-training framework to learn the post, emoji, and word-level representations. The framework consists of two self-supervised tasks: node-level sub-graph contrastive learning and edge-level link reconstruction learning. Both of them are designed to enhance the ability to encode comprehensive semantics among posts, emojis, and words by the general GNN encoder model.

Node-level Sub-graph Contrastive Learning

Task Following Qiu et al. (2020), for each node type, we first sample sub-graphs from the heterogeneous graph via random walk starting from a start node in this node type. Then, we regard the sub-graphs sampled from the same start node as positive pairs, and those from different start nodes as negative pairs. Note that the start nodes of a negative pair should be of the same node type. Finally, we apply the contrastive learning technique to train the GNN encoder by minimizing the InfoNCE loss (Oord et al., 2018):

$$\mathcal{L}_{\mathcal{X}} = -\log \frac{\exp(\mathbf{q}^{\top} \mathbf{k}_{+}/\tau)}{\sum_{i=1}^K \exp(\mathbf{q}^{\top} \mathbf{k}_i/\tau)} \quad (1)$$

where \mathcal{X} denotes a node set and $\mathcal{X} \in \{\mathcal{T}, \mathcal{E}, \mathcal{W}\}$, \mathbf{q} and \mathbf{k}_{+} are the representations of two sub-graphs sampled from the same node $x^q \in \mathcal{X}$, and $\{\mathbf{k}_0, \mathbf{k}_1, \dots, \mathbf{k}_K\}$ is representations collection of other nodes except node x^q , which are generated by the GNN encoder f , denoted by $\mathbf{q} = f(x^q)$ and $\mathbf{k} = f(x^k)$. In this way, the GNN encoder is guided to enhance the similarities (and dissimilarities) between positive (and negative) instances.

Edge-level Link Reconstruction Learning Task

Another task focuses on the relationships among different node types. The motivation is that the representations of semantically similar textual content and emojis should also have higher similarity, even though text and emojis might come from the different distributions. Based on the graph encoder f , we additionally use a score predictor that outputs the

inner product of two nodes' representations as their similarity. If the similarity value exceeds a certain threshold, a edge between the text and emoji nodes is predicted to exist.

To construct training samples, we first sample some existing edges in the heterogeneous graph as positive examples and add an equal number of randomly sampled non-existing edges as negative examples. During training, the model is optimized using a cross-entropy loss to learn representations that maximize the similarity for positive examples and minimize the similarity for negative examples. The binary cross entropy loss for positive edges and negative sampled edges is computed as:

$$\mathcal{L}_+ = -\frac{1}{|\mathcal{P}|} \sum_{(i,j) \in \mathcal{P}} \log p_\theta(y_{i,j} = 1 | z_i, z_j) \quad (2)$$

$$\mathcal{L}_- = -\frac{1}{|\mathcal{N}|} \sum_{(i,j) \in \mathcal{N}} \log (1 - p_\theta(y_{i,j} = 1 | z_i, z_j)) \quad (3)$$

$$\mathcal{L} = \mathcal{L}_+ + \mathcal{L}_- \quad (4)$$

Where \mathcal{P} is the set of positive edges, \mathcal{N} is the set of negative edges, and $p_\theta(y_{i,j} = 1 | z_i, z_j)$ is the probability that the edge (i, j) exists.

For each edge type, we make the training processes separately, resulting in three sub-tasks, *i.e.*, emoji reconstruction, important word identification and emoji-word position reconstruction.

3.3 Downstream Tasks

To evaluate our proposed framework, we apply our emoji embeddings to two downstream tasks: popularity prediction and sentiment analysis. For all downstream tasks, the task input is a social-media post $P = \{T_p; E_p\}$, where $T_p = \{w_{p1}, w_{p2}, \dots, w_{pm}\}$ is the text in the post, w_{pi} means the i -th word in the post text T_p , and m indicates the number of words in T_p . $E_p = \{e_{p1}, e_{p2}, \dots, e_{pn}\}$ denotes the emojis in P , e_j is the j -th emoji, and n represents the number of emojis in E_p . The representations of each word w_{pi} and emoji e_{pj} can be initialized to $h_{w_{pi}} \in R^d$ and $h_{e_{pj}} \in R^d$ via our graph pre-training framework.² We first sample a sub-graph of this node from the heterogeneous graph via a random walk starting from the node, and then input the sub-graph into our GNN encoder model to get its representation,

²If a word or an emoji (in a downstream post) does not appear during pre-training, it will be discarded to avoid the out-of-vocabulary problem.

which also serves as the embedding of the node. This embedding is then used for downstream tasks.

Popularity Prediction & Sentiment Analysis

These two tasks are both multi-class classification tasks, which take P as input and output the popularity/sentiment prediction (assuming there are c classes). The process can be formulated as:

$$H_{ep} = \sum_{i=1}^n h_{e_{pi}} \quad (5)$$

$$H_{eo} = h_t E^T E \quad (6)$$

$$y_p = W_{pop}(h_t \oplus H_{ep} \oplus H_{eo})^T \quad (7)$$

where $E \in R^{K \times d}$ is the matrix embedding of the emoji set \mathcal{E} (includes pre-trained embeddings of all emojis). $h_t \in R^d$ is the embedding of the post text T_p , which is obtained from pre-trained language model backbones. $W_{pop} \in R^{c \times 3d}$ is the trainable parameters. $y_p \in R^c$ denote the predicted probability over all classes.

For objective, we choose the multi-class cross entropy, and we use the weighted labels to mitigate the imbalance of labels. The weight of labels is the inverse of their count.

$$L_{pop} = -\sum_{i=1}^c w_i \hat{y}_i \log(y_{pi}) \quad (8)$$

where w_i is the label weight for the i -th category, \hat{y}_i is the target and y_{pi} is the predicted probabilities for the i -th category.

4 Experiment

In this section, we first introduce the implementation details of our proposed pre-training framework (§ 4.1), and then evaluate our framework on two downstream tasks, *i.e.*, popularity prediction (§ 4.2) and sentiment classification (§ 4.3).

4.1 Experimental setup

During pre-training, we use Adam for optimization with a learning rate of 0.005, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1 \times 10^8$, weight decay of $1e-4$, learning rate warm-up over the first 7,500 steps, and linear decay of the learning rate after 7,500 steps. Gradient norm clipping is applied with range $[-1, 1]$. For MoCo (He et al., 2020), we use a mini-batch size of 32, dictionary size of 16,384, and momentum m of 0.999, the temperature τ is set as 0.07, and we adopt GIN (Xu et al., 2018) with 2 layers and 768 hidden units each layer as

| Metric | Mean | Std | Min | .25 | .50 | .75 | Max |
|------------------|-------|--------|------|-------|-------|-------|----------|
| Liked Count | 799.0 | 5198.5 | 0.0 | 66.0 | 148.0 | 334.0 | 357163.0 |
| Collected Count | 468.3 | 3431.5 | 0.0 | 25.0 | 75.0 | 189.0 | 281984.0 |
| Comments Count | 46.3 | 140.7 | 0.0 | 9.0 | 22.0 | 46.0 | 8652.0 |
| Emoji Count/Post | 9.3 | 8.4 | 2.0 | 4.0 | 7.0 | 12.0 | 316.0 |
| Post Length | 391.0 | 258.2 | 13.0 | 191.0 | 335.0 | 535.0 | 18500.0 |

Table 1: Dataset Description Statistic.

| Backbone | BERT | GPT2 | RoBERTa | OPT-1.3B | LIAMA2-7B |
|--|---------------|---------------|---------------|---------------|---------------|
| <i>F1 score (Weighted) - Prediction of # of likes</i> | | | | | |
| Remove | 0.3801 | 0.3203 | 0.3453 | 0.3602 | 0.4452 |
| Keep V1 | 0.3697 | 0.3659 | 0.3702 | 0.3470 | 0.4489 |
| Keep V2 | 0.3793 | 0.3535 | 0.3867 | 0.3578 | 0.4524 |
| Translate | 0.3820 | 0.3664 | 0.3442 | 0.3440 | 0.4362 |
| Emoji2vec | 0.4357 | 0.3279 | 0.3421 | 0.3771 | 0.4878 |
| Emoji Co-occurrence | 0.3153 | 0.4009 | 0.4136 | 0.3999 | 0.4912 |
| Our model | 0.4487 | 0.4084 | 0.4444 | 0.4174 | 0.5120 |
| <i>F1 score (Weighted) - Prediction of # of collects</i> | | | | | |
| Remove | 0.4060 | 0.3930 | 0.3924 | 0.4073 | 0.4132 |
| Keep V1 | 0.3989 | 0.3887 | 0.3979 | 0.3890 | 0.4534 |
| Keep V2 | 0.4003 | 0.3840 | 0.3966 | 0.3910 | 0.4487 |
| Translate | 0.3973 | 0.3929 | 0.3865 | 0.3881 | 0.4134 |
| Emoji2vec | 0.4377 | 0.3949 | 0.3927 | 0.4082 | 0.4695 |
| Emoji Co-occurrence | 0.3419 | 0.4307 | 0.4374 | 0.4414 | 0.4487 |
| Our model | 0.4387 | 0.4552 | 0.4680 | 0.4580 | 0.4776 |
| <i>F1 score (Weighted) - Prediction of # of comments</i> | | | | | |
| Remove | 0.3816 | 0.3861 | 0.3854 | 0.3987 | 0.4134 |
| Keep V1 | 0.3689 | 0.3556 | 0.3712 | 0.3806 | 0.4383 |
| Keep V2 | 0.3863 | 0.3872 | 0.3769 | 0.3842 | 0.4313 |
| Translate | 0.3662 | 0.3805 | 0.3803 | 0.3881 | 0.4134 |
| Emoji2vec | 0.4317 | 0.3789 | 0.3893 | 0.3969 | 0.4631 |
| Emoji Co-occurrence | 0.2968 | 0.3891 | 0.4005 | 0.3894 | 0.4407 |
| Our model | 0.4353 | 0.4077 | 0.4256 | 0.4066 | 0.4695 |

Table 2: Experimental results on popularity prediction.

our encoders. The GNN encoder trained in two types of pre-training tasks is then used to generate embeddings of post, emoji and text for downstream tasks.

4.2 Popularity Prediction

Dataset Our first dataset mainly comes from the Chinese social media platform Xiaohongshu³. We collect over 4 million Xiaohongshu posts through web crawling, and all the information are desensitized. In addition to the content of the post, other information mainly includes the author’s Xiaohongshu ID, post title (text information), number of likes, number of collects, number of comments, and publishing time. The user information mainly includes the number of followers, the total number of likes, and whether the account is an official certified account. Note that this Xiaohongshu dataset

will not be public due to ethical considerations (e.g., the post might contain individuals’ privacy). Our popularity label is derived from the “liked count”, “collected count”, and “comments count” fields in the Xiaohongshu dataset. Given the continuous nature of this data, to simplify it into a categorical task, we cluster them based on the distribution trend of popularity according to the data percentile of [0-50), [50-80), and [80-100], resulting in three categories (low, medium, and high). Thus, the ratio of category labels is low:medium:high = 5:3:2, indicating the imbalanced labels in the data. Given this, we use the weighted labels to mitigate the imbalance issue. The weights of the labels are the inverses of their counts, as described in Eq. 8. The statistic of dataset is shown in table 1.

Compared Baselines To evaluate the performance of our pre-training framework, we also employ two types of emoji processing methods as baselines:

³<https://www.xiaohongshu.com/>

- **Preprocessing methods:** We use the traditional emoji preprocessing methods introduced in section 2 as baselines: (1) *Remove* emojis: we use vectors of zeroes as the embedding of emojis to eliminate the information contained in them. (2) *Keep* emojis: we use vectors of numbers randomly generated from Gaussian distribution as the embedding of emojis, meaning that they are different tokens but the model has no prior information on emojis. We generated two versions of embeddings to avoid randomness, which are denoted as *Keep V1* and *Keep V2*. (3) *Translate* emojis: we translate emojis into their descriptions (e.g., replace “🚗” with the word “car”), and then use our backbone LLMs to encode emojis. In this way, emojis are treated just as normal words.
- **Emoji representation learning method:** These baseline model utilize the *Emoji2vec* (Eisner et al., 2016) and the *Emoji Co-occurrence* (Illendula and Yedulla, 2018) methods to obtain emojis’ embedding. For those emojis not included in the model, we initiate them to vectors of zeroes.

In all popularity prediction methods, the embedding of text in the post T_p is encoded by corresponding different backbones (i.e., BERT (Devlin et al., 2018), GPT2 (Radford et al., 2019), RoBERTa (Liu et al., 2019), OPT-1.3B (Zhang et al., 2022) and Llama2-7B (Touvron et al., 2023)). With regard to baseline models, we do not use DeepMoji (Felbo et al., 2017; Park et al., 2018) and SEntiMoji (Chen et al., 2019a) for they simply use emojis as annotations and do not learn the representation of emojis. Furthermore, the majority of emojis are represented by Unicode characters that LLMs cannot recognize. Therefore, the translate method mentioned above is essentially an enhanced approach of directly inputting the original text containing emojis into an LLM. Consequently, we did not include the latter in the baseline.

Results & Analyses As the dataset is quite imbalanced on each label, we use the *weighted F1 score* as an evaluation metric. To get a robust result, we run each task 10 times with different random initiations and summarize its average metrics.

As illustrated in Table 2, our method outperforms all four baselines on all three sub-types of popularity prediction tasks. This finding suggests that emojis offer extra information for popularity prediction by comparing our model with the *remove* baseline. Besides, traditional methods gen-

erally fail or have a minor effect on extracting information from emojis. When the emoji embedding is initiated by Gaussian random noise, the F1 score even decreases. This is because no improvement in the prediction performance can be attributed to the absence of semantic information in Gaussian noise. The *translate* baseline utilizes LLMs to extract emoji information by translating emojis into words and then encoding them. Different from Gaussian noise, emoji tokens contain valuable semantics. However, due to there is only a very small part of emojis can be translated, we can infer that incorporating emoji information results in nearly no improvement in the F1 score. *Emoji2vec* enhances the extraction of emoji information and achieves better performance compared to *translate*. Our method shows the highest F1 score among all models, indicating its strong capacity for extracting information from emojis. Detailed precision and recall results are shown in appendix A.

To further demonstrate the superior performance of our model over baseline models on downstream tasks, we conducted two-sample t-tests between our model and each baseline model. The result shows that our model significantly outperforms baseline models at the 0.05 significance level. Detailed results of the hypothesis tests are shown in Table 3.

To intuitively observe the difference among these methods for extraction of the emoji embedding, we conduct the t-SNE (van der Maaten and Hinton, 2008) visualization for the generated emoji embeddings. As shown in Figure 2, the embeddings extracted by other models are randomly distributed in the 2D embedding space, whereas our method effectively groups similar emojis together. For example, emojis representing happiness are situated in the upper left area, emojis denoting sad emotions are located in the upper right area, number emojis are clustered together, and object emojis populate the lower region.

Ablation Study The node-level sub-graph contrastive learning task is separately conducted on different types of nodes, and thus includes three sub-tasks, i.e., post-node, word-node and emoji-node contrastive learning. The edge-level link reconstruction also consists of three sub-tasks, i.e., post-emoji, word-post and emoji-word link reconstruction.

To evaluate the effectiveness of these sub-tasks, we conduct ablations by removing each of them in

| Backbone | BERT | GPT2 | RoBERTa | OPT-1.3B | LIAMA2-7B |
|--|--------|--------|---------|----------|-----------|
| <i>F1 score (Weighted) - Prediction of # of likes</i> | | | | | |
| v.s. Remove | 0.0013 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| v.s. Keep V1 | 0.0344 | 0.0023 | 0.0000 | 0.0000 | 0.0000 |
| v.s. Keep V2 | 0.0000 | 0.0015 | 0.0000 | 0.0002 | 0.0000 |
| v.s. Translate | 0.0000 | 0.0003 | 0.0000 | 0.0001 | 0.0000 |
| v.s. Emoji2vec | 0.0462 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| <i>F1 score (Weighted) - Prediction of # of comments</i> | | | | | |
| v.s. Remove | 0.0416 | 0.0000 | 0.0000 | 0.0000 | 0.0014 |
| v.s. Keep V1 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| v.s. Keep V2 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| v.s. Translate | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| v.s. Emoji2vec | 0.0693 | 0.0000 | 0.0000 | 0.0000 | 0.0014 |
| <i>F1 score (Weighted) - Prediction of # of collects</i> | | | | | |
| v.s. Remove | 0.0257 | 0.0000 | 0.0000 | 0.0087 | 0.0052 |
| v.s. Keep V1 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| v.s. Keep V2 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| v.s. Translate | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| v.s. Emoji2vec | 0.0482 | 0.0000 | 0.0000 | 0.0329 | 0.0020 |

Table 3: Results of t-test. Each score represents the average p-value for t-test.

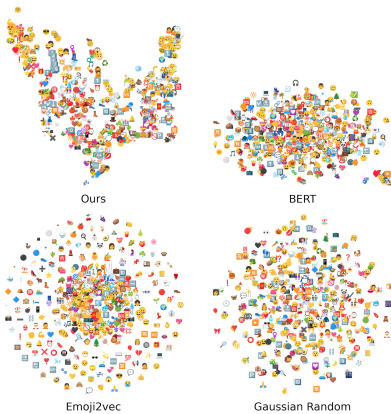


Figure 2: The t-SNE visualization results of emoji embeddings from four models.

our pre-training framework, and evaluate the model performance on popularity prediction (number of likes). As shown in Table 6, the performance of our original framework exceeds every other variant model, indicating that our model gain information from every part of pre-training tasks. In conclusion, the ablation study shows that Emoji representation learning can benefit from all the pre-training sub-tasks, leading to enhanced prediction performance.

4.3 Sentiment Classification

Dataset The dataset used for the sentiment classification task, is a public dataset consisting of Twitter

| Model | F1 score (weighted) |
|---------------------|---------------------|
| Remove | 0.5446 |
| Keep V1 | 0.4336 |
| Keep V2 | 0.4521 |
| Translate | 0.4758 |
| Emoji2vec | 0.5507 |
| Emoji Co-occurrence | 0.5297 |
| SEntiMoji | 0.5512 |
| DeepMoji | 0.5294 |
| Our model | 0.5679 |

Table 4: Experimental results on sentiment classification.

posts generated by authors of Emoji2vec (Eisner et al., 2016), which is also used in their experiments on Emoji2vec. This dataset inherently includes three types of sentiment labels for each post: negative, neutral, and positive. During data preprocessing, we filtered out tweets without emojis from the dataset, separated emojis and text for the rest, and then used BERT to obtain post embeddings.

Compared Baselines To evaluate the performance of our pre-training framework, we employ the same baselines as those in section 4.2. Additionally, we also add SEntiMoji and DeepMoji in our baseline models. Although they do not generate emoji embeddings which can be applied in various downstream tasks, they still serve as strong baselines in sentiment classification task.

| Text | Prediction | Ground Truth |
|--|------------|--------------|
| Maybe got me remembering that shit. Talking about scams. 😞 | Negative | Positive |
| I'm kinda spoiled. 🤔😘👉 | Positive | Negative |
| I miss you too. 🥰👉👉👉👉👉 | Positive | Negative |

Table 5: Some bad cases in sentiments analysis.

| Model | F1 score (weighted) |
|--|---------------------|
| Original framework | 0.4691 |
| - Node level contrastive learning: Post | 0.4529 |
| - Node level contrastive learning: Word | 0.4497 |
| - Node level contrastive learning: Emoji | 0.4434 |
| - Edge level link reconstruction: Post - Emoji | 0.446 |
| - Edge level link reconstruction: Word - Post | 0.4507 |
| - Edge level link reconstruction: Emoji - Word | 0.4401 |

Table 6: Ablation study on node level and edge level tasks. Full model outperforms all other models, meaning that every part of our model is useful.

Results & Analyses As shown in Table 4, our method outperforms all baselines on the sentiment classification task with three categories, achieving a F1 score of 0.5679. SEntiMoji and Emoji2vec performs well, while other baseline models fail to extract emoji information in the posts.

Bad Cases Study In this bad case study of sentiment analysis task, which are shown in Table 5, we observe discrepancies between the model’s predictions and the actual sentiments. These instances highlight several critical challenges.

- **Understanding context and culture:** In the first case, where our method predicted the sentiment as Negative while the ground truth was Positive, the instance likely represents a form of sarcasm or humor. However, in a Chinese context, the emoji 😞 may also carry the connotation of tears of helplessness, pointing towards a complexity involving both the understanding of context and the diverse meanings emojis can have across different linguistic cultures.
- **Ambiguity of emojis and complexity of emotions:** In both the second and third challenging scenarios, the sentences feature a mix of emojis, each imbued with either positive or negative emotional undertones. This blend introduces an intricate layer to the overall sentiment conveyed, making the emotional semantics of the sentence more complex. Although humans may find it relatively straightforward to analyze and understand the sentiment of such sentences, models might struggle to make accurate determinations in these nuanced situations.

5 Conclusion

In this paper, we propose a novel framework for joint pre-training on text and emojis. To model the relationship between posts, words and emojis, we first construct a heterogeneous graph comprising three node types corresponding to each element. Subsequently, we propose a subgraph-level pre-training framework for representation learning, including node-level graph contrastive learning and edge-level link reconstruction learning. Our experimental results on the two downstream tasks, *i.e.*, popularity prediction and sentiment analysis, demonstrate that these pre-training tasks could leverage the acquired embeddings for downstream applications, and our approach yields substantial improvements over baseline methods.

Limitations

We discuss the limitations and how we can potentially address them in this section.

Limitations in Emoji Modalities: While our proposed model could handle the majority of emoji representations, including Unicode encoding and text-based descriptions, it currently lacks the capability to process emoji in image format. This limitation is noteworthy, especially considering the prevalent use of image-based emojis on certain social media platforms. Our future endeavors involve exploring multimodal approaches to enhance our model’s versatility.

Data Cleansing Requirements: The efficacy of our model hinges on accurate emoji tokenization

in data preprocessing. Under certain extreme conditions, such as processing an extensive volume of emojis that resist correct tokenization, the model's performance may be compromised. Addressing these challenges remains a focus for future improvements.

Ethics Statement

There are no ethics-related issues in this paper. The Xiaohongshu data is used only for scientific research purposes, and will not be public. The Twitter data and other related resources in this work are open-source and commonly-used by many existing work.

Acknowledgements

This work was supported in part by NSFC (62206056, 92270121). Additionally, we are grateful to Yang Yang for providing the necessary resources that made this research possible. Finally, we would like to thank the anonymous reviewers for their thoughtful comments and suggestions that helped improve the quality of this paper.

References

- Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saied Safaei, Elizabeth D Trippe, Juan B Gutierrez, and Krys Kochut. 2017. A brief survey of text mining: Classification, clustering and extraction techniques. *arXiv preprint arXiv:1707.02919*.
- Zhenpeng Chen, Yanbin Cao, Xuan Lu, Qiaozhu Mei, and Xuanzhe Liu. 2019a. [Sentimoji: an emoji-powered learning approach for sentiment analysis in software engineering](#). In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE 2019*, page 841–852, New York, NY, USA. Association for Computing Machinery.
- Zhenpeng Chen, Xuan Lu, Sheng Shen, Qiaozhu Mei, Ziniu Hu, and Xuanzhe Liu. 2019b. [Emoji-powered representation learning for cross-lingual sentiment classification](#).
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bosnjak, and Sebastian Riedel. 2016. emoji2vec: Learning emoji representations from their description. In *SocialNLP@EMNLP*.
- Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738.
- Andreas Hotho, Andreas Nürnberger, and Gerhard Paaß. 2005. A brief survey of text mining. *Journal for Language Technology and Computational Linguistics*, 20(1):19–62.
- Anurag Illendula and Manish Reddy Yedulla. 2018. Learning emoji embeddings using emoji co-occurrence network graph. *arXiv preprint arXiv:1806.07785*.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
- Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.
- Ji Ho Park, Peng Xu, and Pascale Fung. 2018. [PlusEmo2Vec at SemEval-2018 task 1: Exploiting emotion knowledge from emoji and #hashtags](#). In *Proceedings of the 12th International Workshop on Semantic Evaluation*, pages 264–272, New Orleans, Louisiana. Association for Computational Linguistics.
- CS Pavan Kumar and LD Dhinesh Babu. 2019. Novel text preprocessing framework for sentiment analysis. In *Smart Intelligent Computing and Applications: Proceedings of the Second International Conference on SCI 2018, Volume 2*, pages 309–317. Springer.
- Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. 2020. Gcc: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1150–1160.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Gopendra Vikram Singh, Dushyant Singh Chauhan, Mauajama Firdaus, Asif Ekbal, and Pushpak Bhat-tacharyya. 2022. [Are emoji, sentiment, and emotion Friends? a multi-task learning for emoji, sentiment, and emotion analysis](#). In *Proceedings of the 36th Pacific Asia Conference on Language, Information and Computation*, pages 166–174, Manila, Philippines. De La Salle University.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. [Llama 2: Open foundation and fine-tuned chat models](#).

Laurens van der Maaten and Geoffrey Hinton. 2008. [Visualizing data using t-sne](#). *Journal of Machine Learning Research*, 9(86):2579–2605.

Sanjaya Wijeratne, Lakshika Balasuriya, Amit Sheth, and Derek Doran. 2017. [Emojinet: An open service and api for emoji sense discovery](#).

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*.

L. O.U. Yinxia, Yue Zhang, L. I. Fei, Tao Qian, and J. I. Donghong. 2020. [Emoji-based sentiment analysis using attention networks](#). *ACM Transactions on Asian and Low-Resource Language Information Processing*, 19.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Sirui Zhao, Hongyu Jiang, Hanqing Tao, Rui Zha, Kun Zhang, Tong Xu, and Enhong Chen. 2023. Pedm: A multi-task learning model for persona-aware emoji-embedded dialogue generation. *ACM Transactions on Multimedia Computing, Communications and Applications*, 19(3s):1–21.

A Full Results of Popularity Prediction

Table 7 and Table 8 show the recall and precision of our model and baseline models, which also verify the superiority of our model.

| Backbone | BERT | GPT2 | RoBERTa | OPT-1.3B | LIAMA2-7B |
|--|---------------|---------------|----------------|-----------------|------------------|
| <i>Recall (Weighted) - Prediction of # of likes</i> | | | | | |
| Keep V1 | 0.4056 | 0.3912 | 0.4017 | 0.3649 | 0.4650 |
| Keep V2 | 0.4180 | 0.3766 | 0.4251 | 0.3698 | 0.4634 |
| Remove | 0.4375 | 0.3879 | 0.3703 | 0.3875 | 0.4952 |
| Translate | 0.4060 | 0.4343 | 0.4237 | 0.3807 | 0.4588 |
| Emoji2vec | 0.4338 | 0.3871 | 0.4044 | 0.3788 | 0.4884 |
| Emoji Co-occurrence | 0.3970 | 0.4047 | 0.4253 | 0.4183 | 0.5001 |
| Our model | 0.4464 | 0.4109 | 0.4480 | 0.4190 | 0.5150 |
| <i>Recall (Weighted) - Prediction of # of collects</i> | | | | | |
| Keep V1 | 0.4259 | 0.4158 | 0.4145 | 0.4223 | 0.4588 |
| Keep V2 | 0.4101 | 0.4034 | 0.4127 | 0.4077 | 0.4572 |
| Remove | 0.4314 | 0.3949 | 0.3952 | 0.4019 | 0.4610 |
| Translate | 0.4092 | 0.4196 | 0.4046 | 0.4031 | 0.4653 |
| Emoji2vec | 0.4338 | 0.3993 | 0.3973 | 0.4043 | 0.4710 |
| Emoji Co-occurrence | 0.4009 | 0.4424 | 0.4471 | 0.4452 | 0.4648 |
| Our model | 0.4342 | 0.4566 | 0.4656 | 0.4556 | 0.4763 |
| <i>Recall (Weighted) - Prediction of # of comments</i> | | | | | |
| Keep V1 | 0.4003 | 0.3976 | 0.3860 | 0.3989 | 0.4454 |
| Keep V2 | 0.4087 | 0.4110 | 0.4111 | 0.4008 | 0.4422 |
| Remove | 0.4282 | 0.3987 | 0.3970 | 0.3933 | 0.4625 |
| Translate | 0.3935 | 0.4164 | 0.3975 | 0.4031 | 0.4653 |
| Emoji2vec | 0.4319 | 0.4021 | 0.3989 | 0.3975 | 0.4621 |
| Emoji Co-occurrence | 0.3810 | 0.4137 | 0.4120 | 0.4087 | 0.4531 |
| Our model | 0.4344 | 0.4212 | 0.4322 | 0.4126 | 0.4717 |

Table 7: Recall of experiments on popularity prediction.

| Backbone | BERT | GPT2 | RoBERTa | OPT-1.3B | LIAMA2-7B |
|---|---------------|---------------|----------------|-----------------|------------------|
| <i>Precision (Weighted) - Prediction of # of likes</i> | | | | | |
| Keep V1 | 0.3947 | 0.3948 | 0.3805 | 0.3979 | 0.4710 |
| Keep V2 | 0.4160 | 0.3933 | 0.3790 | 0.3963 | 0.4754 |
| Remove | 0.4766 | 0.3872 | 0.4246 | 0.4421 | 0.5180 |
| Translate | 0.4137 | 0.3963 | 0.3435 | 0.3957 | 0.4442 |
| Emoji2vec | 0.4755 | 0.3923 | 0.3659 | 0.4412 | 0.5103 |
| Emoji Co-occurrence | 0.2839 | 0.4117 | 0.4270 | 0.4139 | 0.5008 |
| Our model | 0.4808 | 0.4248 | 0.4603 | 0.4350 | 0.5207 |
| <i>Precision (Weighted) - Prediction of # of collects</i> | | | | | |
| Keep V1 | 0.4130 | 0.3937 | 0.3965 | 0.3941 | 0.4621 |
| Keep V2 | 0.4190 | 0.4147 | 0.3967 | 0.3999 | 0.4598 |
| Remove | 0.4518 | 0.4102 | 0.4256 | 0.4223 | 0.4715 |
| Translate | 0.4161 | 0.3990 | 0.3948 | 0.4018 | 0.4172 |
| Emoji2vec | 0.4531 | 0.4099 | 0.4236 | 0.4206 | 0.4779 |
| Emoji Co-occurrence | 0.3601 | 0.4333 | 0.4579 | 0.4506 | 0.4523 |
| Our model | 0.4525 | 0.4587 | 0.4804 | 0.4645 | 0.4842 |
| <i>Precision (Weighted) - Prediction of # of comments</i> | | | | | |
| Keep V1 | 0.3950 | 0.3648 | 0.3909 | 0.3989 | 0.4388 |
| Keep V2 | 0.3945 | 0.4005 | 0.3746 | 0.4080 | 0.4343 |
| Remove | 0.4424 | 0.4003 | 0.4204 | 0.4107 | 0.4713 |
| Translate | 0.4176 | 0.3977 | 0.4071 | 0.4018 | 0.4172 |
| Emoji2vec | 0.4419 | 0.3965 | 0.4209 | 0.4075 | 0.4716 |
| Emoji Co-occurrence | 0.3045 | 0.3878 | 0.4008 | 0.3892 | 0.4441 |
| Our model | 0.4450 | 0.4043 | 0.4254 | 0.4061 | 0.4732 |

Table 8: Precision of experiments on popularity prediction.