What Has Been Enhanced in my Knowledge-Enhanced Language Model?

Yifan Hou[1], **Guoji Fu**[2], **Mrinmaya Sachan**[1]
[1]ETH Zürich, [2]Southern University of Science and Technology
[1]{yifan.hou, mrinmaya.sachan}@inf.ethz.ch, [2]11749236@mail.sustech.edu.cn

Abstract

A number of knowledge integration (KI) methods have recently been proposed to incorporate external knowledge into pretrained language models (LMs). Even though knowledge-enhanced LMs outperform base LMs on knowledge-intensive tasks, the inner-workings of these KI methods are not well-understood. For instance, it is unclear which knowledge is effectively integrated into knowledge-enhanced LMs and which is not; and if such integration leads to catastrophic forgetting of already learned knowledge. We show that existing model interpretation methods such as linear probes and prompts have some key limitations in answering these questions. We revisit KI from an information-theoretic view and propose a new theoretically sound probe called *Graph Convolution Simulator* (GCS) for KI interpretation. GCS uses graph attention on the corresponding knowledge graph for interpretation. In our experiments we verify that GCS can provide reasonable interpretation results for two well-known knowledge-enhanced LMs: ERNIE and K-Adapter. We also find that only a marginal amount of knowledge is successfully integrated in these models, and simply increasing the size of the KI corpus may not lead to better knowledge-enhanced LMs.[1]

1 Introduction

Pretrained language models (LMs) have become the backbone of NLP. Recent work has shown that linguistic knowledge is captured quite well in LMs (Liu et al., 2019a; Jawahar et al., 2019). However, LMs are much worse at capturing factual knowledge about the world (Petroni et al., 2019; Wang et al., 2021b). This has led to the development of a number of knowledge integration (KI) methods which integrate external knowledge from knowledge graphs (KGs) into LMs, leading to knowledge-enhanced language models such as *ERNIE* (Zhang et al., 2019) and *K-Adapters* (Wang et al., 2021a).

Even though enhanced LMs perform better on knowledge-intensive tasks, there is little understanding of where this improvement comes from. Which factual knowledge is successfully integrated into the LM, and which kind of knowledge is not, is not well-understood. As new knowledge is integrated in LMs, old knowledge could be *catastrophically forgotten* (Kirkpatrick et al., 2016). KI could also lead to a situation called *catastrophic remembering* (Kaushik et al., 2021), where the old knowledge could prevent the integration of new knowledge. Our understanding about these issues is also limited.

An intuitive way to understand the KI process could be to probe for factual knowledge in the base LM and the enhanced LM respectively, and compare their results for interpretation. However, we find that because of the high variance of classifier-based probes for KG prediction (Menon and Elkan, 2011; Li et al., 2016) and the significant human effort required to verbalize knowledge graph facts into templates (Petroni et al., 2019; Shin et al., 2020), we cannot easily extend existing probe methods to interpret knowledge-integration in LMs (§3).

In this paper, we revisit the KI process and formulate it with an information-theoretic view (§4). We measure the factual knowledge encoded in the LM by the mutual information (MI) between the model representations and the KG (Hou and Sachan, 2021). Now, the integration and forgetting of factual knowledge can be measured using the difference in MI between the KG and the base LM and the MI between the KG and the enhanced LM. Based on this idea, we theoretically derive a transformation composed of graph convolutions to simulate and interpret this change in MI, leading to our proposed probe, Graph Convolution Simulator (GCS) (§5). The interpretation mechanism in GCS is quite different from existing probe methods as

[1]Our code, demo, and instructions of the usage can be found in https://github.com/yifan-h/GCS_KI

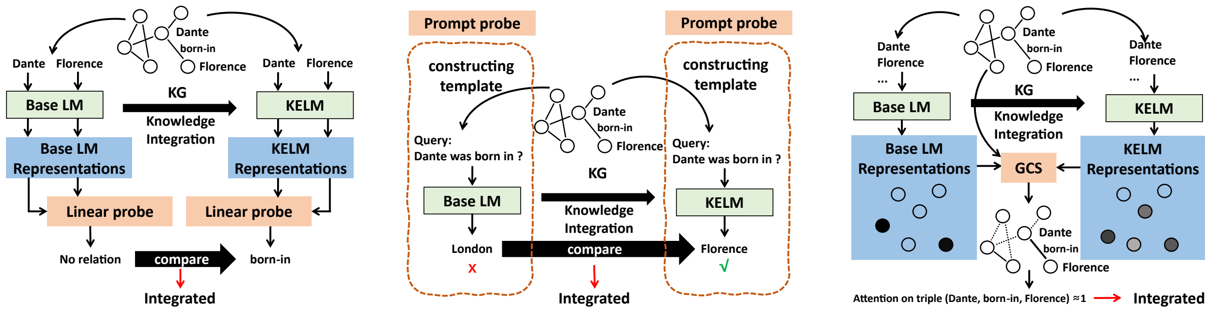


Figure 1: Adaptations of two existing probe methods and GCS to interpret knowledge integration in language models. Probes from left to right are: linear probe, prompt-based probe, and GCS. We can find that only GCS does not have the *compare* operation, which can avoid introducing extra noise of interpretation results.

shown in Figure 1. GCS uses a graph attention layer on the KG to simulate the MI change. Then, it interprets KI from the information flow on the KG using graph attention coefficients.

In our experiments (§6), we verify that GCS can provide reasonable KI interpretation results. We show that: (a) results of GCS have significantly smaller variance compared to the linear probe, (b) dropping knowledge identified as *non-integrated* by GCS does not affect enhanced LMs' performance, and (c) enhanced LMs perform better on samples that include knowledge identified as *integrated* by GCS. In particular, we use GCS to understand two well-known knowledge-enhanced LMs: *ERNIE* (Zhang et al., 2019) and *K-Adapter* (Wang et al., 2021a). Our findings are listed as follows.

- Both of them only integrate little new knowledge (i.e., less than 30% knowledge triples are successfully integrated). ERNIE is better at integrating triples with high degree in KGs, while K-Adapter integrates triples with low-degree entities well.

- In our qualitative study, we find that enhanced LMs do not integrate numbers and temporal knowledge well, where only less than 0.01% triples are successfully integrated.

- Finally, we find that there is no positive relationship between KI corpus size and KI quality. This suggests that merely building larger corpus would not be enough, highlighting the need for more fundamental advances in KI.

2 Preliminaries

KI methods. There are several approaches for KI. KI in LMs can be implemented by aligning phrases in text to entities (Peters et al., 2019; Zhang et al., 2019) or triples (Liu et al., 2020; Wang et al., 2021a) and incorporating corresponding entity or the triple embeddings in the LM. KI methods also include modifications to the Transformer architecture (Peters et al., 2019; Liu et al., 2020), verbalizing knowledge triples and using data augmentation for finetuning (Agarwal et al., 2021), and designing objective functions that predict the factual knowledge (Yao et al., 2019; Wang et al., 2021a).

Knowledge graphs. We assume that factual knowledge for integration can be formulated as a KG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where nodes $v_i \in \mathcal{V}$ represent entities, and edges in $\mathcal{E}$ represent relations between them. Let $\mathcal{N}_{v_i}$ denote the set of neighbors of node v_i, and t_i denote the entity label corresponding to the node v_i. Further, let $\boldsymbol{x}_i = \mathrm{LM}(t_i)$ denote the entity (label) representations given by a LM[2], and $\boldsymbol{X} \in \mathbb{R}^{|\mathcal{V}|\times d}$ denote a matrix formed by stacking all entity representations $\boldsymbol{x}_i \in \mathbb{R}^d$. In this paper, we only consider nodes and edges in the KG and ignore other side information such as relation weights, directions and labels[3].

3 Unsuitability of Existing Probes for KI

Classifier probes (Alain and Bengio, 2017; Hewitt and Liang, 2019) and prompt-based probes (e.g. the LAMA probe) (Petroni et al., 2019; Shin et al., 2020) are typically used to test for various kinds of knowledge in LMs. Classifier probes train simple (usually linear) classifiers to predict the linguistic property of interest and the probe accuracy is used for interpretation (Ribeiro et al., 2016; Hewitt and Manning, 2019). However, simple classifiers are

[2] We represent each entity as the average of its word(-piece) embeddings given by the LM as Hewitt and Manning (2019).

[3] Note that the relation label follows the power-law distribution, and existing probe methods also can not support it well as discussed in Appendix B

not powerful enough to make reliable predictions about large KGs (Menon and Elkan, 2011; Li et al., 2016).[4] Moreover, linear probes are also unable to provide reasonable insights for LMs as they suffer from high variance. If we use them to probe two LMs and compare their results for KI interpretation, variance of interpretation would further increase. We provide empirical evidence later in §6.1.1.

Prompting is another popular way to understand what factual knowledge LMs know. Prompts are designed to let LMs solve text infilling problems, and the prompt output is then used for interpretation (Petroni et al., 2019). However, people have to manually design templates for factual knowledge to be probed[5], and the quality of templates is vital in the overall prompt accuracy (Jiang et al., 2021; Li et al., 2022). As KI methods often use large KGs for integration, it would be infeasible to write a large number of templates for all triples in KGs. To address these issues, we introduce the GCS model and its theoretical motivation behind.

4 Knowledge Integration Understanding

First, we revisit KI in LMs by formulating it in an information-theoretic way. Then, we construct transformations to simulate and interpret the KI process. Notations are summarized in Appendix A.

4.1 Knowledge Integration Definition

We measure knowledge in LMs using mutual information (MI) between the knowledge and the LM representations (Hou and Sachan, 2021). We assume that the local graph $\mathcal{G}(v_i)$ contains all factual knowledge regarding on v_i, and successfully integrated knowledge should be reflected in the entity representations. Let $\mathbf{x}$ be a random variable that takes values ranging over all possible entity representations of a LM[6], and $\mathbf{g}$ be a random variable that ranges over all possible corresponding local structures $\mathcal{G}(v_i)$. Mutual information $\mathrm{MI}(\mathbf{x};\mathbf{g})$ can be used to measure the amount of information in $\mathbf{g}$ contained in $\mathbf{x}$ as

$$\mathrm{MI}(\mathbf{x};\mathbf{g}) = D_{KL}(\mathbb{P}_{\mathbf{xg}}||\mathbb{P}_{\mathbf{x}} \otimes \mathbb{P}_{\mathbf{g}}),$$

which is equivalent to the Kullback-Leibler (KL) divergence between the joint distribution $\mathbb{P}_{\mathbf{xg}}$ and the product of the marginal distributions $\mathbb{P}_{\mathbf{x}} \otimes \mathbb{P}_{\mathbf{g}}$. Next, we present a formal definition of KI.

Definition 4.1 (Knowledge Integration)**.** Let $\mathbf{x}$, $\mathbf{h}$ denote random variables for entity representations in the base LM and the enhanced LM, respectively. The KI process can be formulated as a function f such that $\mathbf{h} = f(\mathbf{x},\mathbf{g})$. Consequently, we assume that the knowledge change during KI can be measured using MI by: $\mathrm{MI}(\mathbf{x};\mathbf{g}) \rightarrow \mathrm{MI}(\mathbf{h};\mathbf{g})$.

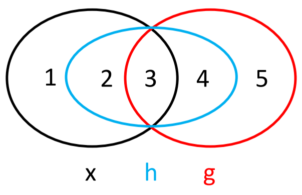


Figure 2: Venn diagram of KI. $\mathbf{x}$ and $\mathbf{h}$ are random variables for entity representations of base LM and enhanced LM. $\mathbf{g}$ is the random variable for the local graph structure.

Definition 4.1 can be intuitively visualized by Figure 2. Ideally, if most knowledge is successfully integrated without much forgetting of the old knowledge, regions 2 and 4 are large. We have $\mathrm{MI}(\mathbf{h};\mathbf{x}) \approx \mathrm{MI}(\mathbf{x};\mathbf{x})$, and $\mathrm{MI}(\mathbf{h};\mathbf{g}) \approx \mathrm{MI}(\mathbf{g};\mathbf{g})$. If little new knowledge has been integrated, i.e., catastrophic remembering happens, region 4 is small, and we have $\mathrm{MI}(\mathbf{h};\mathbf{g}) \approx \mathrm{MI}(\mathbf{x};\mathbf{g})$. Similarly, if catastrophic forgetting happens, region 2 is small and we have $\mathrm{MI}(\mathbf{h};\mathbf{x}) \approx \mathrm{MI}(\mathbf{x};\mathbf{g})$.

4.2 Knowledge Integration Simulation

Note that f in Definition 4.1 shows how KI happens and is an unknown ground-truth transformation that depends on many factors such as the base LM, KI corpus, and KI methods. Thus, we propose an *approximated transformation* f' to fit f such that it can simulate and approximate f with high accuracy (§4.2.1). However, high accuracy cannot promise interpretation. To interpret KI in a fine-grained way, we propose another *interpretable transformation* f'' such that it can be as close to f as f' under the MI measurement (§4.2.2). Figure 3 briefly illustrates the idea. The shown black dashed transformation (i.e., f') can simulate KI with arbitrary accuracy. The solid red lines represent the interpretable transformation (i.e., f'') using graph convolutions, which could promise accuracy with the MI measurement and it is interpretable. Below we introduce details of the two transformations.

4.2.1 Approximated Transformation

Note that samples of $\mathbf{g}$ are non-Euclidean (local graph structures) while samples of $\mathbf{x}$ and $\mathbf{h}$ are

[4]Note that if a more powerful probe model is used, it would learn the task itself instead of probing the task as discussed in (Hewitt and Manning, 2019; Pimentel et al., 2020)

[5]Even if existing prompt methods could learn the template automatically (Shin et al., 2020), the training process could also learn the task and provide unreliable probe results (Jiang et al., 2021; Zhong et al., 2021; Cao et al., 2021).

[6]Here, the set of entity representations $\boldsymbol{X} \in \mathbb{R}^{|\mathcal{V}|\times d}$ can be regarded as empirical samples from $\mathbf{x}$.

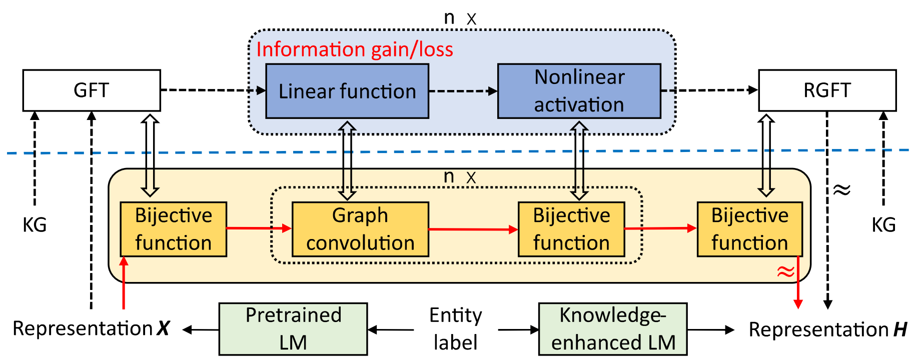


Figure 3: Illustration of the KI simulation. The part above the horizontal blue dashed line represents graph spectral domain (i.e., KG space). GFT and RGFT are graph Fourier transformation and its inverse transformation. Black dashed arrows show the *approximated transformation* (i.e., f'), which can promise the approximation accuracy. Red arrows show the *interpretable transformation* with graph convolutions (i.e., f''), which can promise the accuracy under the MI measurement and interpretability.

vectors. To understand how $\mathbf{x}$ is transformed to $\mathbf{h}$ by integrating $\mathbf{g}$, we first map $\mathbf{x}$ and $\mathbf{h}$ to a new space related to $\mathbf{g}$. Graph Fourier transform (GFT) (Sandryhaila and Moura, 2014) can be used to transform the entity representation $\mathbf{x}$ from the spatial domain to the graph spectral domain (KG space). We denote the transformation of $\mathbf{x}$ to the KG space as $\mathrm{GFT}(\mathbf{x})$, and its inverse transformation as $\mathrm{RGFT}(\mathrm{GFT}(\mathbf{x})) = \mathbf{x}$. Formal definition can be found in Appendix C. Using GFT, we will look at the change from $\mathbf{x}$ to $\mathbf{h}$ in the KG space, and construct an approximated transformation to simulate the KI process there.

Theorem 4.2 (Approximation)**.** *Given a base LM and its enhanced LM, suppose that* $\mathrm{MI}(\mathbf{x};\mathbf{g}) < \mathrm{MI}(\mathbf{h};\mathbf{g})$. *Then, for any* $\epsilon > 0$, *there exists an* n*-layer neural network* $\mathrm{NN}^n(\cdot)$ *such that*

$$|f(\mathbf{x},\mathbf{g}) - \mathrm{RGFT}(\mathrm{NN}^n(\mathrm{GFT}(\mathbf{x})))| < \epsilon.$$

The proof can be found in Appendix D. Theorem 4.2 shows that there exists an approximated transformation composed of GFT and a neural network that can simulate f with arbitrary accuracy. However, the transformation f' cannot provide specific insights about spatial samples. For example, it cannot show which set of knowledge triples contribute to KI. Thus, to interpret KI, we develop a new transformation which can promise both accuracy and interpretability.

4.2.2 Interpretable Transformation

In order to achieve an interpretable transformation with high accuracy, we make use of the invariance property of MI (Kraskov et al., 2004), i.e., the property that bijective functions would not change the MI. If we can change the metric in Theorem 4.2 from $L1$ norm to MI, and replace operations in $f'(\mathbf{x},\mathbf{g})$ that change MI by other equivalent and interpretable operations, we can obtain an interpretable transformation to simulate KI.

Graph convolutions (Defferrard et al., 2016) are often used to model the relational information in KGs by filters (i.e., kernels), where entities aggregate information from their neighbors and pass the information based on the KG structure. Let $\mathrm{GC}(\cdot)$ denote the convolution operation on $\mathcal{G}$. Formal definition of GC can be found in Appendix C. If we run graph convolutions with attention (Velickovic et al., 2018), the information flow on graphs can be indicated by attention coefficients (Zheng et al., 2020; Fu et al., 2020), which can be used for interpretation. Thus, we propose an interpretable transformation to simulate KI process using graph convolutions with attention mechanism.

Theorem 4.3 (Interpretation)**.** *Given a base LM and its enhanced LM, let* $\mathrm{MI}(\mathbf{x};\mathbf{g}) < \mathrm{MI}(\mathbf{h};\mathbf{g})$. *Denote that* $f'(\mathbf{x},\mathbf{g}) = \mathrm{RGFT}(\mathrm{NN}^n(\mathrm{GFT}(\mathbf{x})))$. *Let* $\mathrm{MLP}_b(\cdot)$ *denote the bijective MLP layer and* $\mathrm{GC}(\cdot)$ *denote the graph convolution on KG* $\mathcal{G}$. *There exists* $f''(\mathbf{x},\mathbf{g}) = \mathrm{MLP}_b(\mathrm{GC}^n(\mathrm{MLP}_b(\mathbf{x})))$, *where* GC^n *is composed of* n *repeat components as* $\mathrm{MLP}_b(\mathrm{GC}(\cdot))$, *such that*

$$\mathrm{MI}(f''(\mathbf{x},\mathbf{g}); f(\mathbf{x},\mathbf{g})) = \mathrm{MI}(f'(\mathbf{x},\mathbf{g}); f(\mathbf{x},\mathbf{g})).$$

The proof can be found in Appendix E. Note that the equality above becomes approximate equality when graph filters are approximated filters. For example, GCN (Kipf and Welling, 2017) as well as its variants (e.g., GAT (Velickovic et al., 2018)) runs graph convolutions using localized first-order approximated filters. Theorem 4.3 shows that we can use graph convolution operations to gain interpretability without the loss of MI. Assuming that MI can be used to measure knowledge in LMs as defined before, the interpretable transformation in Theorem 4.3 can promise accuracy as well.

4.3 Knowledge Integration Interpretation

As shown in Theorem 4.3, f'' is composed of n graph convolution operations and $n+2$ bijective functions (bijective MLPs). Since MI does not change with bijective functions, the information change (i.e., $\mathrm{MI}(\mathbf{x};\mathbf{g}) \rightarrow \mathrm{MI}(\mathbf{h};\mathbf{g})$) can only happen in graph convolutions in $f''(\mathbf{x},\mathbf{g})$. We then use graph attention coefficients in the graph convolutions to interpret how the information flows in a KG. Based on the information flow, we can interpret the integration of knowledge triples.

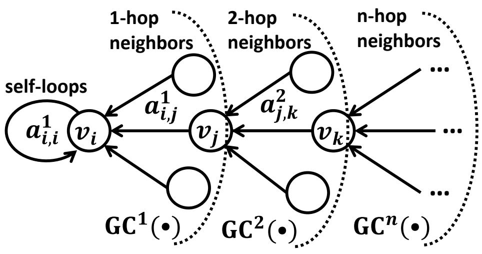


Figure 4: Information flow on knowledge graph with respect to the entity v_i. Information from n-hop neighbors is aggregated with n-th graph convolution. Here, the aggregation number n corresponds to the number of layers of the neural network $\mathrm{NN}^n(\cdot)$ in Theorem 4.3.

Figure 4 illustrates the information flow on KG with respect to v_i. Given a stacking of n graph convolutions, the i-th graph convolution aggregates information from the i-hop neighbors. After the transformation, v_i keeps $a^1_{i,i}$ $(0 < a^1_{i,i} < 1)$ of its original information, and aggregates $1 - a^1_{i,i}$ of information from its n-hop neighbor entities. For example, if $n = 2$, the 2-hop neighbor v_k contributes $a^1_{i,j} \cdot a^2_{j,k}$ proportion of information to v_i.

We now use the information flow for interpretation, where attention coefficients on self-loops can be used to show if catastrophic remembering and catastrophic forgetting happened. For example, given entity v_i, if $a^1_{i,i} \approx 0$, it means entity v_i did not keep any information of itself from the base LM in the enhanced LM, which means that catastrophic forgetting happens to v_i. Similarly, if $a^1_{i,i} \approx 1$, it means that catastrophic remembering happened to v_i. Attention coefficients on the triples can be used to show if they are captured or not during KI. For example in Figure 4, if $a^1_{i,j} \approx 1$, it means that from the base LM to the enhanced LM, entity v_i became more closely associated with v_j. This indicates that the knowledge triple (v_i, r, v_j) is newly integrated during KI. In our experiment, we regard knowledge triples with $a^1_{i,j} > 0.1$ as integrated triples, and others as non-integrated ones. Entities with $a^1_{i,i} < 0.1$ imply that catastrophic forgetting happened, and correspondingly, $a^1_{i,i} > 0.9$ implies that catastrophic remembering happened.[7]

5 Graph Convolution Simulator

So far, we have shown that there exists an interpretable transformation that can simulate KI theoretically. Next, we describe how to optimize the transformation (e.g., the MLP layers, and graph convolution layers) and introduce a practical implementation of our final GCS probe model.

To implement the probe in practice, we make three approximations in GCS: two in the model design and one in optimization. We design experiments in the next section to make sure that GCS works well empirically. The transformation as described in Theorem 4.3 is implemented by bijective MLP layers and graph convolutional layers with attention. We show that if the weight matrix in MLP is a square matrix, the function is bijective. Formal description and proof are in Appendix F. For graph convolutions with attention, we use approximated graph attention filters similar to (Thekumparampil et al., 2018; Velickovic et al., 2018) (Approximation 1). Then, we assume that the knowledge being integrated in the LM can be expressed as triples. In other words, we do not consider multi-hop knowledge triples (e.g., the 2-hop knowledge triple (v_i, r, v_k) in Figure 4) in KI. Thus, we design GCS on the simple condition, where there is only one graph convolution operation (Approximation 2). Therefore, GCS is designed as two bijective MLP layers and one graph attention layer in between as:

$$\mathrm{GCS}_{\theta_1}(\cdot) = \mathrm{MLP}(\mathrm{GC}(\mathrm{MLP}(\cdot))),$$

where $\mathrm{MLP}(\cdot)$ is the bijective MLP layer and $\mathrm{GC}(\cdot)$ is the graph convolutional layer with attention on $\mathcal{G}$. Given an entity v_i and its neighbors $\mathcal{N}_{v_i}$, we can write the graph convolutional layer as[8]:

$$\mathrm{GC}(\boldsymbol{x}_i) = \sigma\left(\sum_{v_j \in \mathcal{N}_{v_i} \cup \{v_i\}} a_{i,j} \boldsymbol{W}^V \boldsymbol{x}_j\right),$$

$$a_{i,j} = \mathrm{softmax}\left(\frac{(\boldsymbol{W}^Q \boldsymbol{x}_i) \cdot (\boldsymbol{W}^K \boldsymbol{x}_j)}{t}\right).$$

[7]Users may choose different thresholds. We heuristically set these thresholds in our analysis for computational reasons.

[8]For notation simplicity, we define $a_{i,j} := a^1_{i,j}$.

Here, $\boldsymbol{x}_i$ is the entity representation of v_i from the base LM. The activation function $\sigma(\cdot)$ is $\mathrm{ELU}(\cdot)$, and $\boldsymbol{W}^V$ is a weight matrix. $a_{i,j}$ is the attention coefficient on the relation that connects v_i and v_j. $\boldsymbol{W}^Q$ and $\boldsymbol{W}^K$ are two parameter matrices in the graph attention. $\mathrm{softmax}(\cdot)$ is the edge-wise softmax function with respect to node v_i. The temperature t is a hyperparameter that controls the attention distribution to be hard or soft.

We optimize GCS using the MI between its outputs and the entity representations from the enhanced LM:

$$\mathcal{L} = -\mathrm{MI}(\mathrm{GCS}_{\theta_1}(\mathbf{x}); \mathbf{h}). \quad (1)$$

In practice, we maximize the compression lemma lower bound of MI instead as introduced in Belghazi et al. (2018). More details can be found in Appendix G.1. In practice, there may be a gap between the ground-truth MI and the compression lemma lower bound, and a stochastic optimization (e.g., Adam (Kingma and Ba, 2015)) may not converge to the optimal point. Thus, GCS may not fit $f''(\mathbf{x}, \mathbf{g})$ perfectly (Approximation 3).

6 Experiments

We begin by reviewing ERNIE (Zhang et al., 2019) and K-Adapter (Wang et al., 2021a). Knowledge is integrated in *entity-wise* manner in ERNIE, and *triple-wise* manner in K-Adapter.

ERNIE. ERNIE integrates knowledge into BERT (Devlin et al., 2019) using a Wikipedia corpus and Wikidata KG. As no alignment is provided between sentences in the Wikipedia corpus and entities in the Wikipedia KG, ERNIE uses TAGME (Ferragina and Scaiella, 2010) to extract entity mentions in sentences and aligns them with corresponding entities in KGs. A new objective is designed for KI in addition to the standard MLM and NSP objectives: alignments in the input text are randomly masked, and the model is asked to select aligned entities from KGs. When ERNIE finds the aligned entity, its KG embedding obtained from Bordes et al. (2013) is integrated into the hidden representations of BERT.

K-Adapter. K-Adapter takes RoBERTa (Liu et al., 2019b) as the base LM and inserts three new layers into RoBERTa to integrate factual knowledge. The final output is concatenated with the output of RoBERTa.[9] During the integration, parameters of RoBERTa are frozen, only parameters in the adapter are updated. K-Adapter uses the T-REx-rc (Elsahar et al., 2018) dataset for KI, which has an alignment of sentences with knowledge triples in Wikidata. For the KI objective, K-Adapter decides whether certain relations exist or not, and classifies relation labels given the aligned sentence.

6.1 GCS Verification

Next, we design a set of experiments to show that GCS can provide reasonable interpretations for KI. We first compare GCS with the linear probe with respect to variance of the interpretation results (§6.1.1). We show that the linear probe does not work in interpreting KI, but GCS can provide stable interpretations. Then, we verify GCS based on the KG used in KI (§6.1.2). We drop knowledge triples that are identified as *non-integrated* by GCS during KI and show that it would not affect the performance of enhanced LMs. Third, we verify GCS based on the downstream tasks (§6.1.3). We show that enhanced LMs perform well on the test samples that contain *integrated* knowledge triples that are identified by GCS, and vice versa.

6.1.1 Variance of interpretation results

As alluded to earlier, linear probes do not work for large-scale factual knowledge interpretation. We support this claim by evaluating the variance of interpretation results. We use the entropy of probe results to test if the interpretation is stable.

Setting. We run a linear probe and GCS to detect if a knowledge triple is *integrated* or *non-integrated* 100 times with different random seeds.[10] We calculate the entropy of these results to estimate the probe variance. If a triple is classified as "unlearned" in the base LM but "learned" in the enhanced LM, we regard it as *integrated* and if it is classified as "unlearned" in both base LM and enhanced LM, we say the triple is *non-integrated*. For GCS, as introduced in §4.3, if the attention coefficient for the triple is larger than 0.1, it is deemed as *integrated*, else not.

Results. A histogram of the entropy of all the entities is shown in Figure 5. For a random guess strategy, the entropy of most knowledge triples is around 1, which means the results are highly unstable. For the linear probe, we find that only 10% of

[9]Note that here we only consider factual knowledge, thus, the linguistic Adapter is not used.

[10]We use 2-fold cross validation for interpretation, where knowledge triples are randomly split into two even parts for training and testing.

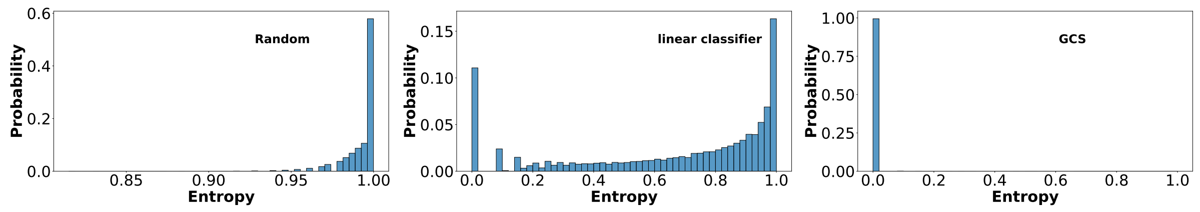


Figure 5: The histogram of entropy for all entities (K-Adapter), where x-axis shows the entropy value and y-axis shows the empirical probability (i.e., frequency) of entities. The random guess strategy is also included for comparison. We can find that the KI interpretation results using linear probes have fairly large variance, which are similar to the results of random guessing. But our GCS can provide stable interpretations.

knowledge triples are interpreted in a stable manner. The other 90% triples have large entropy (their interpretation is similar to random guessing). On the other hand, for GCS, most knowledge triples have stable interpretation results. This shows that GCS can indeed provide more reliable interpretations.

6.1.2 Verification via the KI corpus

As our second verification experiment, we only use the factual knowledge that is identified as *integrated* by GCS to enhance BERT and RoBERTa to get ERNIE (drop-UE) and K-Adapter (drop-UE). Then we judge if the GCS interpretation was reasonable using two downstream tasks, where enhanced LMs outperform base LMs most significantly. If GCS can interpret the KI process well, the performance of the drop-UE versions should be roughly the same as that of ERNIE/K-Adapter.

Setting. This experiment comprises of three steps. First, we use GCS to interpret the KI process in ERNIE and K-Adapter, and identify triples or entities that are integrated successfully. Second, we re-enhance BERT/RoBERTa to get ERNIE (drop-UE) / K-Adapter (drop-UE) only using the entities/triples that are identified as *integrated*. Third, we finetune ERNIE/K-Adapter and their drop-UE versions on two downstream tasks.

After we get our interpretation results, we only keep the KI corpus aligned with the *integrated* knowledge.[11] We finetune the models on two downstream tasks about entity typing: OpenEntity (Choi et al., 2018) and FIGER (Ling et al., 2015). Implementation details of GCS, ERNIE, and K-Adapter can be found in Appendix G.1 and Appendix G.2.

Results. From Table 1, we find that even if we drop a large amount of KI data in this way, the performance of drop-UE versions on entity typing task is roughly the same as original versions. For the OpenEntity dataset, better performance is achieved. For the FIGER dataset, the performance of drop-UE versions is slightly worse. We also report the performance of BERT, RoBERTa, and two drop-IE[12] versions in Figure 6. We find that compared to dropping KI corpus aligned with *integrated* knowledge, dropping KI corpus aligned with *non-integrated* knowledge achieves much better performance. Thus, we verify that GCS provides reasonable interpretations.

Table 1: Performance of ERNIE, K-Adapter and their drop-UE versions on the entity typing downstream tasks. We can find that dropping a large amount of non-integrated knowledge would not affect enhanced LMs' performance much on knowledge-intensive downstream tasks.

Model	OpenEntity			FIGER		
	P	R	F1-Micro	P	R	F1-Micro
ERNIE	**78.24**	68.75	73.19	**77.39**	**65.81**	**71.13**
ERNIE (drop-UE)	78.11	**71.43**	**74.62** ↑	77.38	64.90	70.60 ↓
K-Adapter	**76.63**	75.26	75.94	**67.50**	88.79	**76.69**
K-Adapter (drop-UE)	75.95	**75.95**	**75.95** ↑	67.29	**88.88**	76.59 ↓

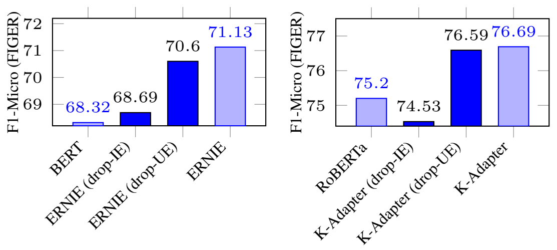


Figure 6: Performance of BERT, RoBERTa, ERNIE, K-Adapter, and their dropped versions on the FIGER dataset. We can find that even if there are negative effects on the performance, they are marginal that can be ignored.

6.1.3 Verification via the downstream task

We use the performance of ERNIE and K-Adapter on downstream tasks to verify GCS. If GCS can reasonably interpret KI, enhanced LMs should perform better on the test set with samples aligned with the *integrated knowledge*, and vice versa.

[11]We only keep 61.72% entity embeddings (obtained by Bordes et al. (2013)) for ERNIE (drop-UE), and 10.09% KI corpus (i.e., natural sentences) for K-Adapter (drop-UE). Detailed statistics are in Table 6 in Appendix H.

[12]The drop-IE versions mean that we drop the subset aligned with integrated factual knowledge identified by GCS and use the rest to enhance BERT and RoBERTa .

Setting. We first align entities in the KI corpus and the OpenEntity dataset based on their *Wikidata Q identifier*.[13] For the entity typing task (OpenEntity dataset), we drop samples in the finetuning test set that aligns with the *integrated* knowledge and *non-integrated* entities (called *w/o-IE* test set and *w/o-UE* test set), and test ERNIE and K-Adapter on the two dropped test sets. Detailed statistics can be found in the Table 7 in Appendix H.

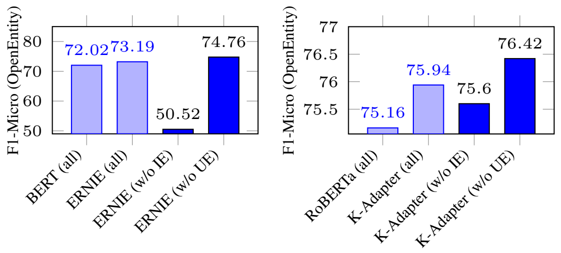


Figure 7: Performance of BERT, RoBERTa, ERNIE, K-Adapter on the OpenEntity dataset for different test sets (original versions and dropped versions). We can find that enhanced LMs perform better on test samples that contain successfully integrated knowledge.

Results. As shown in Figure 7, we find that for ERNIE, the difference is significant. The performance on the test set (w/o-IE) is more than 20 F1 points worse than that on the complete test set (all). For K-Adapter, there is a drop in F1 on the w/o-IE set and increase in F1 on the w/o-UE set (albeit small). We hypothesize that this may be because of the differences in the finetuning objective and the KI objective[14], and because the knowledge integrated in K-Adapter may change during finetuning. These results show that GCS can reasonably interpret which set of knowledge is integrated.

6.2 GCS Findings

After verifying GCS with three set of experiments, we analyze the interpretation results. We find that both ERNIE and K-Adapter integrate only few knowledge triples ($\approx 20 - 30\%$). Detailed results can be found in Figure 11 in Appendix H. Next, we classify the factual knowledge based on its relation types (in terms of their topology type and Wiki data type) and analyze how ERNIE and K-Adapter integrate knowledge with certain relation types.

6.2.1 Analysis via relation topology

We classify relations into three types based on their topology features. Following previous work (Bordes et al., 2013; Tang et al., 2020), we denote relations that connect two leaf nodes (entities) in the KG as $1-1$ relations, and relations that connect two center nodes (entities) in the KG as $N-M$ relations. Others are denoted as $N-1$ relations. An example can be found in Figure 12 in Appendix H. We perform an analysis in terms of different types of relations and report the percentage of successfully integrated entities and triples for ERNIE and K-Adapter. For each relation type, we also present the percentage of connected entities that are catastrophically remembered or forgotten (CR and CF).

[13]https://www.wikidata.org/wiki/Q43649390

[14]K-Adapter integrates knowledge in a triple-wise manner only using a small amount of adapter parameters.

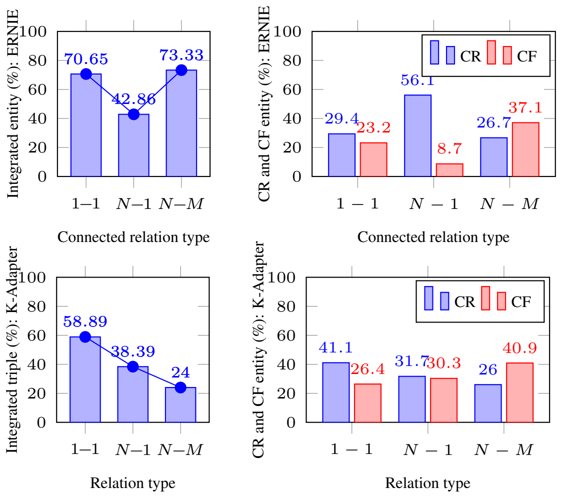


Figure 8: Analysis of KI interpretation results in terms of different relation topology. We can find that the degree of knowledge (types) integration is different for enhanced LMs using different KI methods.

Figure 8 presents the results, and detailed statistics can be found in Table 8 in Appendix H. We find that for ERNIE, entities connected with complex relations (i.e., $N-M$ relations) are captured well. However, K-Adapter shows different behaviors. It captures triples with simple relations (i.e., $1-1$ and $N-1$ relations) well. Note that ERNIE uses graph embeddings provided by the specially designed model (Bordes et al., 2013) for KI, while K-Adapter integrates knowledge into dedicated feed-forward networks called adapters. This implies that structures are not well encoded into classic adapter modules, and we may need a better approach to integrate knowledge with complex structures into neural modules. Besides, we find that for both ERNIE and K-Adapter, CR happens more often to entities in simple structures (i.e., $N-1$ relations), while CF is more common for entities in complex structures (i.e., connected to $N-M$ relations).

6.2.2 Analysis via relation's Wiki features

For further analysis, we select six relations aligned with roughly the same number of sentences in the

Table 2: Analysis of KI interpretations via different relation labels for the T-REx-rc dataset (KI corpus used by K-Adapter). We can find that temporal knowledge is hard to get integrated.

Relation label	T-REx-rc		
	Wiki Count	Wiki data type	Integrated triple
place of birth (LF)	2,850,424	Wikibase item	10.95%
part of (LF)	4,164,470	Wikibase item	17.25%
date of death (TR)	2,637,358	Time	<0.01%
date of birth (TR)	5,294,649	Time	<0.01%
located in the administrative territorial (HF)	10,776,120	Wikibase item	6.13%
country (HF)	14,174,811	Wikibase item	0.12%
Total	-	-	10.09%

T-REx-rc dataset[15] and categorize them into three groups based on the *Wiki Count* and *Wiki data type*[16]: low-frequency (LF) relations, time-related (TR) relations, and high-frequency (HF) relations. From Table 2, we find that even if LF relations have roughly the same Wiki Count as TR relations, the temporal knowledge still cannot be integrated by K-Adapter. We speculate that this is because Transformer encoders do not capture information about time well (Dhingra et al., 2021; Zhou et al., 2021). When comparing LF relations and HF relations, we find that if relations have small Wiki Count, knowledge triples are easily captured.

Table 3: Examples of triples in the T-REx-rc dataset (KI corpus used by K-Adapter) with attention coefficients.

Knowledge triple	Attention coefficient
(Adam Smith, place of birth, Kirkcaldy)	1.079×10^{-1}
(Lake Huron, part of, Great Lakes)	1.742×10^{-1}
(Jean-Jacques Rousseau, date of death, 02 July 1778)	1.729×10^{-25}
(Barack Obama, date of birth, 04 August 1961)	6.827×10^{-31}
(Mauna Kea Observatory, located in the administrative territorial, Hawaii)	6.044×10^{-2}
(China, country, Mahalangur Himal)	1.250×10^{-3}

Randomly picked examples of knowledge triples are given in Table 3. We can find that for TR relations, they connect entities composed of numbers. The poor performance of language models in handling numbers (Wallace et al., 2019) provides an alternative explanation for the observation that K-Adapter does not integrate triples with TR relations. For triples with LF and HF relations, we find that some entities connected to HF relations are very common (e.g, entity "China" is in the complex KG structure) compared to those connected to LF relations. These results are consistent with our findings in Figure 8 that knowledge of popular entities is not integrated.

[15]Statistics are in Table 9 in Appendix H

[16]See the Wikipedia page for more details.

6.2.3 Can we further improve the KI quality?

Finally, we attempt to answer the question: *can we simply improve the quality of KI by increasing the amount of our aligned training corpora?* Intuitively, repeatedly learning a knowledge triple with several aligned sentences could increase the possibility of successful integration of this knowledge.

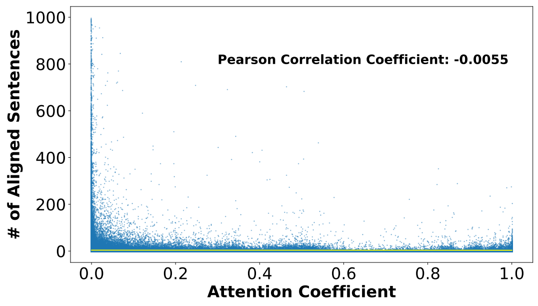


Figure 9: The correlation between the attention coefficient of the knowledge triple and its aligned sentence number in T-REx-rc dataset. We can find that there is no correlation between them, which means simply increasing the KI corpus could not be helpful for better KI quality.

We answer this question by calculating correlation between the attention coefficients (i.e., success ratio of integration) for K-Adapter and number of aligned sentences (i.e., size of KI corpus) for knowledge triples in the T-REx-rc dataset. Surprisingly, we find that this *Pearson correlation* is -0.0055 (Figure 9). This shows that there is no apparent positive relationship between the KI quality and the size of the KI dataset. It suggests that simply increasing the size of the aligned dataset alone may not improve KI and we might need more fundamental advances to push the state-of-the-art in KI.

7 Conclusion

In this paper, through a series of theoretical results, we derived an information-theoretic probe that uses attention over knowledge graphs to interpret the knowledge integration process in LMs. In our experiments, we verified our probe model and used it to understand what knowledge has been integrated in two existing knowledge-enhanced language models, leading to some new findings about these models. We hope that our probe model would aid in better understanding and informed design of knowledge integration approaches for LMs. We have published the code and the demo to help users easily implement GCS for their own knowledge-enhanced LM interpretation.

Limitations

There are some limitations of our work. We simplify the KG without considering relation information such as labels, since existing graph neural networks (e.g., R-GCN (Schlichtkrull et al., 2018)) still cannot handle such large number of imbalanced distributed relations. These can be considered by future works. Besides, GCS only provides a way to interpret the knowledge integration. Once we have an understanding of the it, improving the integration quality still remains challenging.

Reproducibility Statement

We have published the code, the demo, and the interpretation results. The design of GCS can be found in §5. The implementation details about the knowledge integration for K-Adapter and ERNIE can be found in Appendix G.2, and details about GCS can be found in Appendix G.1.

Ethics Statement

While our probe models are not tuned for any specific real-world application, our methods could be used in sensitive contexts such as legal or healthcare settings; and it is essential that any work that builds on our approaches undertake extensive quality-assurance and robustness testing before using it in their setting.

Acknowledgment

We are grateful to the anonymous reviewers for their insightful comments and suggestions, which helped us significantly improve the paper. We also owe many thanks to Shehzaad Dhuliawala and Nico Daheim for their constructive advice on this paper. Yifan Hou is supported by the Swiss Data Science Center PhD Grant (P22-05). We also acknowledge support from an ETH Zurich Research grant (ETH-19 21-1) and a grant from the Swiss National Science Foundation (project #201009) for this work.

References

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2021. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3554–3565, Online. Association for Computational Linguistics.

Guillaume Alain and Yoshua Bengio. 2017. Understanding intermediate layers using linear classifier probes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Arindam Banerjee. 2006. On bayesian bounds. In *ICML*, volume 148 of *ACM International Conference Proceeding Series*, pages 81–88. ACM.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, R. Devon Hjelm, and Aaron C. Courville. 2018. Mutual information neural estimation. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 530–539. PMLR.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Neural Information Processing Systems*, pages 2787–2795.

Ronald Newbold Bracewell and Ronald N Bracewell. 1986. *The Fourier transform and its applications*, volume 31999. McGraw-Hill New York.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2014. Spectral networks and locally connected networks on graphs. In *ICLR*.

Boxi Cao, Hongyu Lin, Xianpei Han, Le Sun, Lingyong Yan, Meng Liao, Tong Xue, and Jin Xu. 2021. Knowledgeable or educated guess? revisiting language models as knowledge bases. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1860–1874, Online. Association for Computational Linguistics.

Zhengdao Chen, Soledad Villar, Lei Chen, and Joan Bruna. 2019. On the equivalence between graph isomorphism testing and function approximation with gnns. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 15868–15876.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 87–96, Melbourne, Australia. Association for Computational Linguistics.

George Cybenko. 1992. Approximation by superpositions of a sigmoidal function. *Math. Control. Signals Syst.*, 5(4):455.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Neural Information Processing Systems*, pages 3837–3845.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2021. Time-aware language models as temporal knowledge bases. *CoRR*, abs/2106.15110.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REx: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Paolo Ferragina and Ugo Scaiella. 2010. TAGME: on-the-fly annotation of short text fragments (by wikipedia entities). In *CIKM*, pages 1625–1628. ACM.

Guoji Fu, Yifan Hou, Jian Zhang, Kaili Ma, Barakeel Fanseu Kamhoua, and James Cheng. 2020. Understanding graph neural networks from graph signal denoising perspectives. *CoRR*, abs/2006.04386.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Yifan Hou and Mrinmaya Sachan. 2021. Bird's eye: Probing for linguistic graph structures with a simple information-theoretic approach. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1844–1859, Online. Association for Computational Linguistics.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Prakhar Kaushik, Alex Gain, Adam Kortylewski, and Alan L. Yuille. 2021. Understanding catastrophic forgetting and remembering in continual learning with optimal relevance mapping. *CoRR*, abs/2102.11343.

Nicolas Keriven and Gabriel Peyré. 2019. Universal invariant and equivariant graph neural networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7090–7099.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*. OpenReview.net.

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2016. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. 2004. Estimating mutual information. *Physical review E*, 69(6):066138.

Bopeng Li, Sougata Chaudhuri, and Ambuj Tewari. 2016. Handling class imbalance in link prediction using learning to rank techniques. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 4226–4227. AAAI Press.

Jiaoda Li, Ryan Cotterell, and Mrinmaya Sachan. 2022. Probing via prompting. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1144–1157, Seattle, United States. Association for Computational Linguistics.

Xiao Ling, Sameer Singh, and Daniel S. Weld. 2015. Design challenges for entity linking. *Transactions of the Association for Computational Linguistics*, 3:315–328.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-BERT: enabling language representation with knowledge graph. In *AAAI*, pages 2901–2908. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Aditya Krishna Menon and Charles Elkan. 2011. Link prediction via matrix factorization. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011, Proceedings, Part II*, volume 6912 of *Lecture Notes in Computer Science*, pages 437–452. Springer.

Ilsang Ohn and Yongdai Kim. 2019. Smooth function approximation by deep neural networks with general activation functions. *Entropy*, 21(7):627.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54, Hong Kong, China. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-theoretic probing for linguistic structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4622, Online. Association for Computational Linguistics.

Marco Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 97–101, San Diego, California. Association for Computational Linguistics.

Aliaksei Sandryhaila and José M. F. Moura. 2014. Discrete signal processing on graphs: Frequency analysis. *IEEE Trans. Signal Process.*, 62(12):3042–3054.

Michael Sejr Schlichtkrull, Nicola De Cao, and Ivan Titov. 2020. Interpreting graph neural networks for NLP with differentiable edge masking. *CoRR*, abs/2010.00577.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Yun Tang, Jing Huang, Guangtao Wang, Xiaodong He, and Bowen Zhou. 2020. Orthogonal relation transforms with graph context modeling for knowledge graph embedding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2713–2722, Online. Association for Computational Linguistics.

Kiran Koshy Thekumparampil, Chong Wang, Sewoong Oh, and Li-Jia Li. 2018. Attention-based graph neural network for semi-supervised learning. *CoRR*, abs/1803.03735.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR*. OpenReview.net.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP models know numbers? probing numeracy in embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5307–5315, Hong Kong, China. Association for Computational Linguistics.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2021a. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1405–1418, Online. Association for Computational Linguistics.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021b.

KEPLER: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. KG-BERT: BERT for knowledge graph completion. *CoRR*, abs/1909.03193.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

Bo Zheng, Haoyang Wen, Yaobo Liang, Nan Duan, Wanxiang Che, Daxin Jiang, Ming Zhou, and Ting Liu. 2020. Document modeling with graph attention networks for multi-grained machine reading comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6708–6718, Online. Association for Computational Linguistics.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [MASK]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5017–5033, Online. Association for Computational Linguistics.

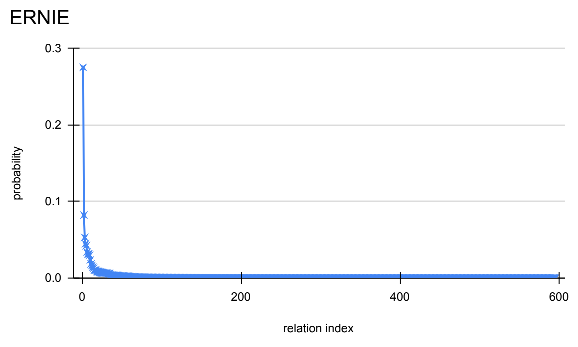
Ben Zhou, Kyle Richardson, Qiang Ning, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2021. Temporal reasoning on implicit events from distant supervision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1361–1371, Online. Association for Computational Linguistics.

A Notations

The notations can be found in the below table.

B Relation Distribution

We present the features of relation label distribution using two real KGs used for integration in ERNIE and K-Adapter, and we briefly illustrate that existing probe methods cannot support the relation label well.



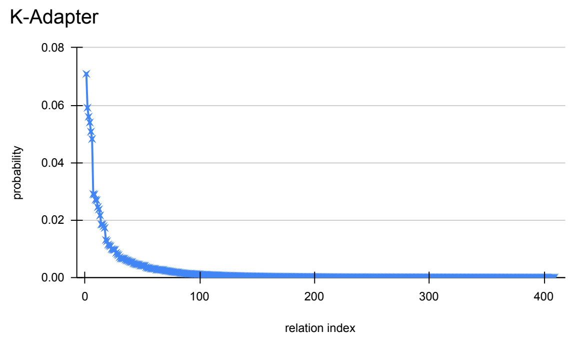


Figure 10: The distribution of relations with respect to number in two KGs used in KI for ERNIE and K-Adapter.

- **The number of relations is large.** The KG used in ERNIE has 594 distinct relations and the KG used in K-Adapter has 410 distinct relations. As for probes, if we differentiate them but use shared parameters such as the linear probe, the probe task is to predict whether the relation exist and if yes, which relation label it is. There is no wonder that using simple linear model cannot handle such difficult task with that large number of labels for classification. Regarding prompt probe, for each KG, the template for each relation should be manually created by human, and it is fairly costly.

  If we use different sets of parameters for different relations like in RGCN (Schlichtkrull et al., 2018). It is hard to implement hundreds or thousands of sets of parameters to analyze KI. For example, in our GCS model, we have to include attention mechanism for interpretation. The number of parameters for attention mechanism is hard to be scaled to $400-600$ times. Even if we can address this technical issue, using such as complex probe model for analysis is also problematic.

- **The distribution of relation number is highly imbalanced.** As shown in Figure 10, we can see that the distribution (i.e., PDF) of relations is very imbalanced. For ERNIE, 10% relations account for 93% edges, and 5 relations (around 1%) account for 50% edges. For K-Adapter, 10% relations account for 78% edges, and 5 relations (around 1%) account for 29% edges. Simply treating relations in different ways in interpretation could also provide problematic results. For example, in the linear probe, the simple linear model could not handle such highly imbalanced labels for classification.

C Formal Definitions of GFT and Graph Convolutions

Formal Definition of GFT. Specifically, given a KG denoted as $\mathcal{G}$, let $\boldsymbol{A} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ be its symmetric adjacency matrix corresponding to $\mathcal{G}$. Let $\boldsymbol{L}_n = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$ denote the normalized Laplacian matrix for $\mathcal{G}$, where $\boldsymbol{D}$ denotes the degree matrix. We do the eigendecomposition for $\boldsymbol{L}_n$ as $\boldsymbol{L}_n = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$, where $\boldsymbol{U}$ is the matrix of eigenvectors ordered by eigenvalues and $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, ..., \lambda_N)$ is the diagonal matrix of eigenvalues. Consider the node feature matrix as $\boldsymbol{X}$. The GFT can be written as $\mathrm{GFT}(\boldsymbol{X}) = \boldsymbol{U}^T\boldsymbol{X}$, and the inverse GFT can be written as $\mathrm{RGFT}(\mathrm{GFT}(\boldsymbol{X})) = \boldsymbol{U}\mathrm{GFT}(\boldsymbol{X}) = \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{X} = \boldsymbol{X}$.

Formal Definition of Graph Convolutions. Graph convolutions (Bruna et al., 2014) can be implemented by filters (i.e., kernels) as g_Θ in the graph spectral domain (i.e., KG space). As the GFT of the convolution of g_Θ and $\boldsymbol{X}$ is the pointwise product of their GFT (Bracewell and Bracewell, 1986), the convolution can be written as

$$\mathrm{GC}(\boldsymbol{X}) = g_\Theta \star \boldsymbol{X} = \mathrm{RGFT}(g_\Theta \cdot \mathrm{GFT}(\boldsymbol{X})).$$

Regarding graph filters, Velickovic et al. (2018) and Thekumparampil et al. (2018) introduce the attention mechanism to them, where the contribution of

Table 4: Notations and their descriptions

Notation	Description
$\mathcal{G}$	The knowledge graph for KI
$\mathcal{V}$	The set of entities/nodes of KG
$\mathcal{E}$	The set of edges of KG
v_i	The entity/node indexed as i in the KG
t_i	The entity label attached on v_i
$\mathrm{LM}(\cdot)$	The language model, where the input is entity text, and the output is its representation
$\mathcal{N}_{v_i}$	The set of neighbors (entities/nodes) connected to v_i
$\mathcal{G}(v_i)$	The local graph structure in terms of v_i
$\mathbf{x}$	The random variable of the entity representation
$\boldsymbol{x}_i$	The entity representations of v_i
$\mathbf{g}$	The random variable of the local graph structure
$\mathrm{MI}(\cdot;\cdot)$	The mutual information between two random variables
$\boldsymbol{A}$	The adjacency matrix of KG
$\|\mathcal{V}\|$	The number of entities/nodes in KG
$\mathbb{R}$	The set of real numbers
$\boldsymbol{I}$	The identity matrix
$\boldsymbol{D}$	The degree matrix of KG
$\boldsymbol{L}_n$	The normalized Laplacian matrix
$\mathrm{diag}(\cdot)$	The diagonalization operation
$\boldsymbol{U}$	The matrix of eigenvectors
$\boldsymbol{\Lambda}$	The diagonal matrix of eigenvalues
λ_i	The i-th eigenvalue
$\boldsymbol{X}$	The set of entity representations in terms of $\mathcal{V}$
C or d	The dimension of entity representations; The number of channels
$\mathrm{GFT}(\cdot)$	The graph Fourier transformation
$\mathrm{RGFT}(\cdot)$	The inverse graph Fourier transformation
g_Θ	The graph filter parameterized by parameter Θ
$\boldsymbol{H}$	The entity representations given by a knowledge-enhanced LM
$\mathbf{h}$	The random variable of the entity representation given by a knowledge-enhanced LM
$f(\cdot,\cdot)$	The mapping that can transform $\mathbf{x}$ to $\mathbf{h}$ with $\mathbf{g}$
ϵ	The error of the approximation
$\mathrm{sigmoid}(\boldsymbol{x})$	The Sigmoid function $\mathrm{sigmoid}(\cdot) = \frac{1}{1+e^{-\boldsymbol{x}}}$
n	The number of layers of the neural network for apporximation
$\boldsymbol{W}$	The weight matrix
$\boldsymbol{x}$	The input vector
$\boldsymbol{b}$	The bias (in the weight matrix)
λ_0'	The minimum eigenvalue of the weight matrix $\boldsymbol{W}$
$\mathrm{MLP}_b(\cdot)$	The bijective MLP function
$\mathrm{GC}(\cdot)$	The graph convolution function with respect to KG $\mathcal{G}$
GCS_{θ_1}	The GCS model parameterized by θ_1
$\mathcal{L}$	The loss function (objective) of the optimization
$\boldsymbol{Z}$	The output of GCS, i.e., set of output entity representations
$\mathbf{z}$	The random variable of the output of GCS
$\sup$	The supremum value
T	A class of functions
$\mathcal{F}$	Any class of functions
Ω	The domain of a function
T_{θ_2}	A class of functions parameterized by θ_2, i.e., neural networks
$\mathbb{P}$	The probability distribution
$\mathbb{P}^{\|\mathcal{V}\|}$	The empirical distribution with $\|\mathcal{V}\|$ samples
$\mathrm{NN}_\sigma(\cdot\|\theta')$	The neural network with activation function $\sigma(\cdot)$ and parameterized by θ'
$\|\boldsymbol{U}\|$	The norm of matrix $\boldsymbol{U}$
$\boldsymbol{A}_n$	The normalized adjacency matrix
$\hat{\boldsymbol{X}}$	The ground-truth entity representations/node features
$\hat{\boldsymbol{X}}^*$	The variable matrix
$\mathbf{Tr}(\cdot)$	The trace of a matrix
ϵ_1, ϵ_2	The error bound of entity representations/node features and adjacency matrix
γ	The Lagrangian multiplier
$p(t)$	The characteristic polynomial for weight matrix $\boldsymbol{W}$
$\det(\cdot)$	The determinant of a matrix

each edge to the convolution can be shown explicitly. Graph attention makes filters more powerful and convolutions more interpretable (Velickovic et al., 2018; Thekumparampil et al., 2018; Fu et al., 2020; Zheng et al., 2020).

D Proof of Theorem 4.2

Proof. As aforementioned, the graph Fourier transformation $\mathrm{GFT}(\cdot)$ and its inverse transformation $\mathrm{RGFT}(\cdot)$ in terms of the KG $\mathcal{G}$ can be written as

$$\mathrm{GFT}(\boldsymbol{X}) = \boldsymbol{U}^T\boldsymbol{X}$$
$$\mathrm{RGFT}(\mathrm{GFT}(\boldsymbol{X})) = \boldsymbol{U}\mathrm{GFT}(\boldsymbol{X}) = \boldsymbol{U}\boldsymbol{U}^T\boldsymbol{X} = \boldsymbol{X}.$$

The second equation can be derived since $\boldsymbol{U}$ is the set of eigenvectors of the normalized Laplacian matrix in terms of $\mathcal{G}$, which is orthogonal.

According to the universal approximation theorem (Cybenko, 1992), in general, we can use one-layer neural networks (arbitrary width) with the sigmoid activation function to fit any functions. Ohn and Kim (2019) bound the approximation with both the width and depth, and supports more activation functions. Based on the conclusion of Ohn and Kim (2019), we know that given a mapping $g'(\cdot)$, for any $\epsilon' > 0$, there exists a neural network parameterized by θ' s.t.

$$|g'(\cdot) - \mathrm{NN}_\sigma(\cdot|\theta')| < \epsilon'.$$

Note that there are some constraints about the input and the model architecture, i.e., layer width. We leave out those details for simplicity since we only focus on the existence.

Since $\mathbf{h}$ is obtained by integrating $\mathbf{g}$ into $\mathbf{x}$, we can simplify the mapping in the graph spectral space by researching on the transformation from $\mathrm{GFT}(\mathbf{x})$ to $\mathrm{GFT}(\mathbf{h})$.[17] Assume the mapping satisfies $g'(\mathrm{GFT}(\mathbf{x})) = \mathrm{GFT}(\mathbf{h})$. Then we have

$$|g'(\mathrm{GFT}(\mathbf{x})) - \mathrm{NN}_\sigma(\mathrm{GFT}(\mathbf{x})|\theta')| < \epsilon'.$$

Consider that we have $f(\mathbf{x}, \mathbf{g}) = \mathbf{h} = \mathrm{RGFT}(g'(\mathrm{GFT}(\mathbf{x})))$. If we assign $\epsilon' = \frac{\epsilon}{|\boldsymbol{U}|} > 0$, we have

$$|\boldsymbol{U}| \cdot |g'(\mathrm{GFT}(\mathbf{x})) - \mathrm{NN}_\sigma(\mathrm{GFT}(\mathbf{x})|\theta')| < \epsilon.$$

Since we know that

$$\boldsymbol{U} \cdot g'(\mathrm{GFT}(\mathbf{x})) = \mathrm{RGFT}(g'(\mathrm{GFT}(\mathbf{x}))) = \mathbf{h} = f(\mathbf{x}, \mathbf{g}),$$

[17]In next proof, we illustrate that the linear transformation in the graph spectral space is graph convolution, which integrates the graph information into entities. Thus, in the graph spectral space, we do not need to regard $\mathbf{g}$ as an input. More formally description can be found in Chen et al. (2019); Keriven and Peyré (2019).

we have

$$|f(\mathbf{x}, \mathbf{g}) - \mathrm{RGFT}(\mathrm{NN}(\mathrm{GFT}(\mathbf{x})))|$$
$$< |\boldsymbol{U}| \cdot |g'(\mathrm{GFT}(\mathbf{x})) - \mathrm{NN}_\sigma(\mathrm{GFT}(\mathbf{x})|\theta')| < \epsilon,$$

where $\mathrm{NN}(\cdot)$ is parameterized by θ' with activation function σ as $\mathrm{NN}_\sigma(\cdot|\theta')$. And without loss of generality, we assume it is composed of n layers. □

E Proof of Theorem 4.3

According to the invariance property of MI (Kraskov et al., 2004), the introduction of bijective functions does not introduce any new information – MI remains unchanged upon the introduction of bijective functions. We know that GFT and RGFT are both bijective (Appendix E.1). We show that nonlinear activation functions in a neural network (e.g., $\mathrm{sigmoid}(\cdot)$) are bijective as well (Appendix E.1). Thus, the MI change in the KI process can only happen in the linear function (Appendix E.2). Based on the convolution theorem (Bracewell and Bracewell, 1986), linear functions in graph spectral domain are graph convolution operations (Sandryhaila and Moura, 2014; Bruna et al., 2014; Kipf and Welling, 2017) (Appendix E.3). Consider that the graph attention can show how the information flow on graphs during the convolution (Zheng et al., 2020; Fu et al., 2020) (Appendix E.4). Thus, we can use graph convolutions in the transformation to interpret the KI process.

Proof. We present the proof with 4 steps below.

E.1 Step 1

$\mathrm{GFT}(\cdot)$, $\mathrm{RGFT}(\cdot)$, and $\mathrm{sigmoid}(\cdot)$ are bijective. Given two entity representations $\boldsymbol{x}_i$, $\boldsymbol{x}_j$ and the matrix of eigenvectors of the KG as $\boldsymbol{U}$, suppose that $\mathrm{GFT}(\boldsymbol{x}_i) = \mathrm{GFT}(\boldsymbol{x}_j)$. Then, we have

$$\boldsymbol{U}^T\boldsymbol{x}_i = \boldsymbol{U}^T\boldsymbol{x}_j.$$

Since $\boldsymbol{U}^T$ are set of eigenvectors and are by definition nonzero, we have

$$\boldsymbol{x}_i = \boldsymbol{x}_j.$$

If $\boldsymbol{x}_i = \boldsymbol{x}_j$, it is easy to get $\mathrm{GFT}(\boldsymbol{x}_i) = \mathrm{GFT}(\boldsymbol{x}_j)$. Thus, graph Fourier transformation is bijective.

As for the nonlinear activation function, since we consider neural networks composed of MLP layers, the activation function is $\mathrm{sigmoid}(\cdot)$ function. It is easy to find that its inverse function is $f(\boldsymbol{y}) = \ln(1 - \frac{1}{\boldsymbol{y}})$. Similarly, we can prove that it is bijective as well.

E.2 Step 2

Information gain and loss can only happen in the linear function in the graph spectral domain. Without the loss of generality, we set the anchor random variable as $\mathbf{g}$. Same result can be derived using any other random variables. Based on the invariance of MI (Kraskov et al., 2004), we have

$$\begin{aligned}\mathrm{MI}(\mathbf{x};\mathbf{g}) &= \mathrm{MI}(\mathrm{GFT}(\mathbf{x});\mathbf{g}),\\ \mathrm{MI}(\mathbf{x};\mathbf{g}) &= \mathrm{MI}(\mathrm{RGFT}(\mathbf{x});\mathbf{g}),\\ \mathrm{MI}(\mathbf{x};\mathbf{g}) &= \mathrm{MI}(\mathrm{sigmoid}(\mathbf{x});\mathbf{g}).\end{aligned} \tag{2}$$

Since we know that

$$\mathrm{MI}(\mathbf{h};\mathbf{g}) - \mathrm{MI}(\mathbf{x};\mathbf{g}) > 0,$$

and the neural network can well approximate the mapping, we have

$$\begin{aligned}&\mathrm{MI}(\mathbf{h};\mathbf{g}) - \mathrm{MI}(\mathbf{x};\mathbf{g})\\ &\cong \mathrm{MI}(\mathrm{RGFT}(\mathrm{NN}(\mathrm{GFT}(\mathbf{x})));\mathbf{g}) - \mathrm{MI}(\mathbf{x};\mathbf{g})\\ &= \mathrm{MI}(\mathrm{NN}(\mathrm{GFT}(\mathbf{x}));\mathbf{g}) - \mathrm{MI}(\mathrm{GFT}(\mathbf{x});\mathbf{g}) > 0.\end{aligned}$$

If we write $\mathrm{NN}(\cdot)$ with n MLP layers as $n \times \sigma(\mathrm{Linear}(\cdot))$, we have

$$\begin{aligned}&\mathrm{MI}(\mathrm{NN}(\mathrm{GFT}(\mathbf{x}));\mathbf{g}) - \mathrm{MI}(\mathrm{GFT}(\mathbf{x});\mathbf{g})\\ &= \mathrm{MI}(n \times \sigma(\mathrm{Linear}(\mathrm{GFT}(\mathbf{x}));\mathbf{g}) - \mathrm{MI}(\mathrm{GFT}(\mathbf{x});\mathbf{g}).\end{aligned}$$

Recursively with equations 2, it is easy to get that MI only changes in the $\mathrm{Linear}(\cdot)$ functions. And if we can show that linear function in the graph spectral domain is the graph convolution operation, we can then easily get that

$$\begin{aligned}&\mathrm{MI}(n \times \sigma(\mathrm{Linear}(\mathrm{GFT}(\mathbf{x}));\mathbf{g})\\ &= \mathrm{MI}(n \times \mathrm{MLP}_b(\mathrm{GC}(\mathrm{MLP}_b(\mathbf{x}));\mathbf{g}).\end{aligned}$$

E.3 Step 3

The linear function in the graph spectral domain is the graph convolution operation. Even if many existing works (Sandryhaila and Moura, 2014; Bruna et al., 2014; Kipf and Welling, 2017) have provided clear descriptions, we simply re-illustrate it under the multi-channel setting. Consider the graph filter in Bruna et al. (2014) as an examaple.

For a linear function $f(\boldsymbol{x}) = \boldsymbol{W} \times \boldsymbol{x}$, its weight matrix $\boldsymbol{W} \in \mathbb{R}^{F\times C}$ is parameterized by $\Theta \in \mathbb{R}^{F\times C}$. If the parameters are not shared for all nodes, the input $\boldsymbol{X} \in \mathbb{R}^{|\mathcal{V}|\times C}$ can be rescaled in $\mathbb{R}^{|\mathcal{V}|\times C\times 1}$, and the weight matrix is $\boldsymbol{W} \in \mathbb{R}^{|\mathcal{V}|\times F\times C}$ parameterized by $\Theta \in \mathbb{R}^{F\times C\times|\mathcal{V}|}$. The output of this linear function is mapped in $\mathbb{R}^{|\mathcal{V}|\times F}$.

Consider the signal in graph convolution, i.e., all $\boldsymbol{x}$ in $\boldsymbol{X} \in \mathbb{R}^{|\mathcal{V}|\times C}$. Since parameters are not shared (Bruna et al., 2014), for one graph filter, the parameters in g_Θ is in $\mathbb{R}^{C\times|\mathcal{V}|\times|\mathcal{V}|}$ that is parameterized by $\Theta \in \mathbb{R}^{C\times|\mathcal{V}|}$ with simple diagonalization. If we have F different graph filters for the convolution, g_Θ is in $\mathbb{R}^{F\times C\times|\mathcal{V}|\times|\mathcal{V}|}$ that is parameterized by $\Theta \in \mathbb{R}^{F\times C\times|\mathcal{V}|}$. Here, the graph Fourier transformation of $\boldsymbol{X}$ is $\mathrm{GFT}(\boldsymbol{X}) \in \mathbb{R}^{|\mathcal{V}|\times C}$, which can be rescaled in $\mathbb{R}^{1\times|\mathcal{V}|\times C\times 1}$ with simple diagonalization. The output is in $\mathbb{R}^{F\times|\mathcal{V}|\times|\mathcal{V}|\times 1}$. Note that since the parameters in the graph filter is diagonalized, we can rescale the output in $\mathbb{R}^{|\mathcal{V}|\times F}$.

If we regard the weight matrix $\boldsymbol{W}$ as the parameters in the graph filter g_Θ, the input matrix $\boldsymbol{X}$ as the signal, obviously, the linear function in the graph spectral space is the graph convolution operation.

E.4 *Step 4

Graph attention can show how information flows on the graph. Graph attention works as denoising information from neighbors, since it can adaptively learn the optimal weights (i.e., attention coefficients) for different neighbors. If the node features of a neighbor contain much useless information for the center node, the learned weight should be small to denoise that information. It can show how information (i.e., node features) flows among nodes on the KG structure.

Consider a graph signal denoising problem that we aim to extract the ground-truth node features $\hat{\boldsymbol{X}}$ and edge weights $\hat{\boldsymbol{A}}_n$ from a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{A}_n)$ with noise in both node features $\boldsymbol{X}$ and edge weights $\boldsymbol{A}_n$. Here, $\boldsymbol{A}_n$ is the normalized adjacency matrix $\boldsymbol{A}_n = \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$. To this end, we formulate the optimization problem under the assumption that the ground-truth node features $\hat{\boldsymbol{X}}$ are smooth w.r.t the ground-truth adjacency matrix $\hat{\boldsymbol{A}}_n$ and the noise in the graph can be upper-bounded:

$$\begin{aligned}\hat{\boldsymbol{X}}^*, \hat{\boldsymbol{A}}_n^* = &\underset{\hat{\boldsymbol{X}},\hat{\boldsymbol{A}}_n}{\mathrm{argmin}}\mathrm{Tr}\left(\hat{\boldsymbol{X}}\hat{\boldsymbol{L}}_n^T\hat{\boldsymbol{X}}\right)\\ &\text{s.t. } \|\hat{\boldsymbol{X}} - \boldsymbol{X}\|_2^2 \le \epsilon_1,\\ &\quad\ \|\hat{\boldsymbol{A}}_n - \boldsymbol{A}_n\|_2^2 \le \epsilon_2,\end{aligned} \tag{3}$$

where $\hat{\boldsymbol{L}} = \boldsymbol{I} - \hat{\boldsymbol{A}}$, $\epsilon_1, \epsilon_2 \in \mathbb{R}$, are the level of noise in node features and edge weights, respectively. $\mathrm{Tr}(\cdot)$ indicates the trace of a matrix. By Lagrange multipliers methods, we can obtain the solution as following:

$$\hat{\boldsymbol{X}}^* = \frac{\gamma}{1+\gamma}\left(\boldsymbol{I} - \frac{1}{1+\gamma}\hat{\boldsymbol{A}}_n^*\right), \tag{4}$$

$$\hat{\boldsymbol{A}}_n^* = \boldsymbol{A}_n + \sqrt{\epsilon_2}\frac{\hat{\boldsymbol{X}}^*\hat{\boldsymbol{X}}^{*\top}}{\|\hat{\boldsymbol{X}}\|_2^2}, \tag{5}$$

where $\gamma > 0$ is the Lagrangian multiplier. Note that the attention coefficients of GAT (Velickovic et al., 2018) and AGNN (Thekumparampil et al., 2018) are obtained by (without less of generality, we show the results in the first-layer) equation 6 and equation 7, respectively:

$$a_{i,j} = \text{softmax}\left(\text{LReLU}\left(\mathbf{a}^\top\left[\boldsymbol{W}\boldsymbol{X}_i\|\boldsymbol{W}\boldsymbol{X}_j\right]\right)_{j\in\mathcal{N}_i\cup\{i\}}\right), \tag{6}$$

$$a_{i,j} = \text{softmax}\left(\left[\beta\frac{\boldsymbol{H}_i^\top\boldsymbol{H}_j}{\|\boldsymbol{H}_i\|\|\boldsymbol{H}_j\|}\right]_{j\in\mathcal{N}_i\cup\{i\}}\right), \tag{7}$$

where LReLU is the leakyReLU; $\boldsymbol{H} = \text{ReLU}(\boldsymbol{X}\boldsymbol{W})$, $\mathbf{a}$, $\boldsymbol{W}$ in equation 6, and β, $\boldsymbol{W}$ in equation 7 are learnable parameters. The attention coefficents of GAT and AGNN are then used as the weights of aggregating the neighbohood information of nodes. As we can see that equation 5, equation 6, and equation 7 are in a form of measuring the similarity between paired node features. Similar to the denoised edge weights obtained in equation 5, the attention coefficents (i.e. the aggregation weights) between a node and its neighborhoods are proportional to the similarity of their node embeddings. Therefore, the attention coefficients of GAT and AGNN can be regarded as the results of denoised weights on the existing edges in a graph, i.e., the graph attentions are implicitly denoising the edge weights.

In general case, graph attention functions as denoising edge weights. The input is noisy representations and the output is the groundtruth. Attention coefficients show how much distortion is corrected during the convolution operation. For example, if the input representations are also groundtruth, there is no need to fetch information from neighbors to get output. And edge weights will be reduced to 0, i.e., attention coefficients on edges are calculated as 0. If the input representations are very noisy, i.e., much noise are removed, attention coefficients on edges should be large to restore the groundtruth signal. Therefore, in the KI scenario, we can use attention coefficients in graph attention in graph convolution layer to interpret the KI process based on the information flow. As for the CR and CF, equally, we can use the attention coefficients on the self-loop edges for interpretation, such as how much original information is remembered/forgotten.

□

F Bijective MLP

Theorem F.1. *Give an MLP layer denoted as* $\text{MLP}(\boldsymbol{x}) = \text{sigmoid}(\boldsymbol{W}\boldsymbol{x}+\boldsymbol{b})$. *If* $\boldsymbol{W}$ *is a square matrix, there exist a constant* $\lambda_0' > 0$ *that for any* $0 < \epsilon < \lambda_0'$, *the function below is bijective:*

$$\text{MLP}_n(\boldsymbol{x}) = \text{sigmoid}((\boldsymbol{W}-\epsilon\boldsymbol{I})\boldsymbol{x}+\boldsymbol{b}). \tag{8}$$

Proof. We first prove that two bijective function compositions are still bijective. Then, we prove that adding a small noise on MLP weight matrix can make it bijective.

Give two MLP function $f_1(\cdot)$ and $f_2(\cdot)$. Suppose they are injective and suppose $f_1(f_2(\boldsymbol{x})) = f_1(f_2(\boldsymbol{y}))$. Since we know that $f_1(\cdot)$ is injective, we have $f_2(\boldsymbol{x}) = f_2(\boldsymbol{y})$. Similarly, since $f_2(\cdot)$ is injective, we have $\boldsymbol{x} = \boldsymbol{y}$. Thus $f_1(f_2(\cdot))$ is injective. Suppose $f_1(\cdot)$ and $f_2(\cdot)$ are surjective and $\boldsymbol{z} \in C$. Since we know that $f_1(\cdot)$ is surjective, there exists a set of $\boldsymbol{y} \in B$ with $f_1(\boldsymbol{y}) = \boldsymbol{z}$. Similarly, since $f_2(\cdot)$ is surjective, there exists a set of $\boldsymbol{x} \in A$ with $f_2(\boldsymbol{x}) = \boldsymbol{y}$. Then, we have $\boldsymbol{z} = f_1(f_2(\boldsymbol{x}))$ and so $\boldsymbol{z}$ is onto $f_1(f_2(\cdot))$. Thus, $f_1(f_2(\cdot))$ is surjective. Therefore, if $f_1(\cdot)$ and $f_2(\cdot)$ are bijective, $f_1(f_2(\cdot))$ is also bijective.

To prove that the special MLP is bijective, consider an MLP function as

$$\text{MLP}(\boldsymbol{x}) = \sigma(\boldsymbol{W}\boldsymbol{x}+\boldsymbol{b}),$$

where $\boldsymbol{W} \in \mathbb{R}^{C\times C}$ is the weight matrix and $\boldsymbol{b} \in \mathbb{R}^C$ is the bias. Let

$$p(t) = \prod_{i=1}^{C}(\lambda_i' - t)$$

be the characteristic polynomial for weight matrix $\boldsymbol{W}$. Here λ_i' are eigenvalues of matrix $\boldsymbol{W}$. Without loss of generality, let $|\lambda_0'| = \min_i |\lambda_i'|$. Then, we know that for any constant $0 < \epsilon < |\lambda_0'|$, we have

$$\det(\boldsymbol{W}-\epsilon\boldsymbol{I}) = p(\epsilon) \neq 0.$$

Thus, if the perturbation ϵ is small enough, the perturbed matrix $\boldsymbol{W}' = \boldsymbol{W} - \epsilon\boldsymbol{I}$ is nonsingular. Consider the fact that the nonlinear activation function $\sigma(\cdot)$ is sigmoid$(\cdot)$ function, which is bijective. Therefore, the special MLP function $\text{MLP}_n(\cdot)$ is bijective.

Note that in practice, we use the floating-point arithmetic. Consider the float accuracy. Small errors from the float approximation can be regarded as the constant ϵ, and in most cases, it satisfies the assumption $0 < \epsilon < |\lambda'_0|$. Thus, we can regard MLPs with square weight matrices in practice as bijective functions.

$\square$

G Implementation Details

G.1 GCS

In practice, GCS is composed of 3 layers: bijective MLP layer, graph convolutional layer, and another bijective MLP layer. As for bijective MLP layers, since weight matrices in them are square matrices, the dimension would remain unchanged: 768 for ERNIE and 1024 for K-Adapter. The nonlinear activation functions are set as $\mathrm{ELU}(\cdot)$ function, which is also bijective. The learning rate is set as $1e^{-3}$, and the dropout rate of the two MLP layers is 0.2.

Regarding the graph attention, to make sure interpretation results are stable, we apply multi-head attention mechanism, where the number of attention head is set as 8. Entity representations are first embedded into a space with the dimension as 64. Then, the embedded representations are used to calculate the attention coefficients. Note that since the purpose is to simulate and interpret the KI process, we do not split datasets for KI. Considering that GCS model is very simple for large KGs, overfitting is unlikely to happen. Thus, we optimize GCS for the whole datasets. Specifically, for K-Adapter, the whole KG is used for optimization, and results are used for interpretation. And for ERNIE, since the KG is very large, we sample a small subgraph with $1,344,393$ entities and $3,240,272$ triples for optimization (see Table 5), and then implement the optimized GCS on the whole KG for interpretation.

The objective function of optimizing GCS can be reconstruction loss minimization or MI maximization. In this paper, we all select MI maximization as the objective. Note that users can use other objectives such as the reconstruction loss minimization. Regarding the MI maximization, we optimize MI (equation 1) by maximizing the compression lemma lower bound (Banerjee, 2006) as in Belghazi et al. (2018). The inputs of GCS are $\boldsymbol{X}$, and let the output be denoted by $\boldsymbol{Z}$. We can regard $\boldsymbol{Z}$ and $\boldsymbol{H}$ as empirical samples of random variables $\mathbf{z}$ and $\mathbf{h}$. Thus, we have:

$$\mathrm{MI}(\mathbf{z};\mathbf{h}) \geq \sup_{T\in\mathcal{F}} \mathbb{E}_{\mathbb{P}_{\mathbf{zh}}}[T] - \log(\mathbb{E}_{\mathbb{P}_{\mathbf{z}}\otimes\mathbb{P}_{\mathbf{h}}}[e^{T}]). \quad (9)$$

Here, $\mathcal{F}$ can be any class of functions $T:\Omega\to\mathbb{R}$ satisfying certain integrability constraints (Belghazi et al., 2018). $\mathbb{P}_{\mathbf{zh}}$ represents the joint distribution of $\mathbf{z}$ and $\mathbf{h}$, and $\mathbb{P}_{\mathbf{z}}\otimes\mathbb{P}_{\mathbf{h}}$ represents the product of their marginal distributions. In practice, we let $\mathcal{F} = \{T_{\theta_2}\}$ be the set of functions parameterized by a neural network, and optimize it by stochastic gradient descent. Then, the objective function can be rephrased as

$$\max_{\theta_1,\theta_2} \left(\mathbb{E}_{\mathbb{P}^{|\mathcal{V}|}_{\mathbf{z},\mathbf{h}}}[T_{\theta_2}] - \log\left(\mathbb{E}_{\mathbb{P}^{|\mathcal{V}|}_{\mathbf{z}}\otimes\mathbb{P}^{|\mathcal{V}|}_{\mathbf{h}}}[e^{T_{\theta_2}}]\right)\right), \quad (10)$$
$$\text{where } \mathbf{z} = \mathrm{GCS}_{\theta_1}(\mathbf{x}).$$

In equation 10, $\mathbb{P}^{|\mathcal{V}|}_{\mathbf{z}}$ represents the empirical distribution of $\mathbf{z}$, i.e., $\boldsymbol{Z}$. If the KG is very large, we can optimize the network by sampling a small subgraph of the KG. In practice, we simply add two extra MLPs layers to GCS for MI maximization as (Belghazi et al., 2018). The added two MLP layers may not be bijective, where the dimension would be first reduced to 64, then to 1 for MI maximization. The nonlinear activation functions are all set as $\mathrm{ELU}(\cdot)$ function, which is also bijective.

For interpretation, we use the attention coefficients on edges and self-loops to analyze the KI in terms of triples and entities. Different from Schlichtkrull et al. (2020) that specially designs a discrete function to mask edges that are not important, we simply introduce a temperature hyperparameter t and set it as $t = 0.1$ to make the attention coefficient distribution hard.[18] Thus, knowledge can be well clustered into learned and unlearned.

G.2 ERNIE and K-Adapter

KI. To ensure that the experiment settings are fair, we set hyperparameters as the default values. For **K-Adapter**, the code and hyperparameters for KI that we use are from the official projects[19] published by the authors (Wang et al., 2021a). The only two differences are that: we use PyTorch *float 32* instead of *float 16* since BERT and RoBERTa that we use are *float32*, and we use 4 NVIDIA Tesla V100 GPUs for KI training. For **ERNIE**, settings are the same. All hyperparameters for KI are set as their default values.[20] Similarly, *float 16* of PyTorch is changed to *float 32*, and we do the integration with 4 NVIDIA Tesla V100 GPUs.

[18]Note that the principle of hyperparameter selection is to maximize the MI, i.e., objective function. Users may select appropriate hyperparameters depending on the situation.

[19]https://github.com/microsoft/K-Adapter

[20]https://github.com/thunlp/ERNIE

Note that the dataset that ERNIE used for KI is Wikipedia, since the code is to fetch latest version of it, the data that we use could be slightly different. Therefore, for both ERNIE and K-Adapter, to ensure the fairness, we reproduce their KI, and report the results of reproduced models instead of results provided in their papers.

Finetuning. As for the downstream tasks, all the hyperparameters are consistent with the official project: either they are given in the project or in the README. In the same way, *float 32* and 4 NVIDIA Tesla V100 GPUs are chosen to make sure that the comparison is fair. Note that for K-Adapter and ERNIE, the best performance for different datasets is achieved in different settings. For example, the best performance for K-Adapter on the OpenEntity dataset is achieved with single GPU, but on the FIGER dataset is achieved with four GPUs. Since we focus on the relative performance and the fairness of the comparison, we run finetuning on 4 NVIDIA Tesla V100 GPUs for all downstream tasks and all LMs (as well as BERT and RoBERTa).

H Additional Statistics

Table 5: Statistics of T-REx-rc and Wikidata. The datasets that K-Adapter and ERNIE use are T-REx-rc and Wikidata.

Statistics / Datasets	# of entities	# of triples	# of aligned sentences	# of entities (optimization)	# of triples (optimization)
T-REx-rc	781,275	1,282,504	5,565,478	-	-
Wikidata	3,275,534	12,849,311	-	1,344,393	3,240,272

Table 6: Drop statistics for the Integration Experiment.

Statistics / Datasets	Percentage of integrated entities	Percentage of integrated triples	# of aligned sentences/entity embeddings (integrated knowledge)
T-REx-rc	-	28.86%	561,687 out of 5,565,478
Wikidata	61.72%	-	2,240,260 out of 3,275,534

Table 7: Performance change of K-Adapter and ERNIE on the OpenEntity dataset with different test sets.

Model (Test set)	OpenEntity			
	Left test set	P	R	ΔF1-Micro
K-Adapter (w/o IE)	37.44%	− 0.33	− 0.37	− 0.35
K-Adapter (w/o UE)	64.46%	− 0.18	+ 1.12	+ 0.47
ERNIE (w/o IE)	27.28%	− 18.20	− 25.14	− 22.67
ERNIE (w/o UE)	66.87%	− 0.31	+ 3.08	+ 1.57

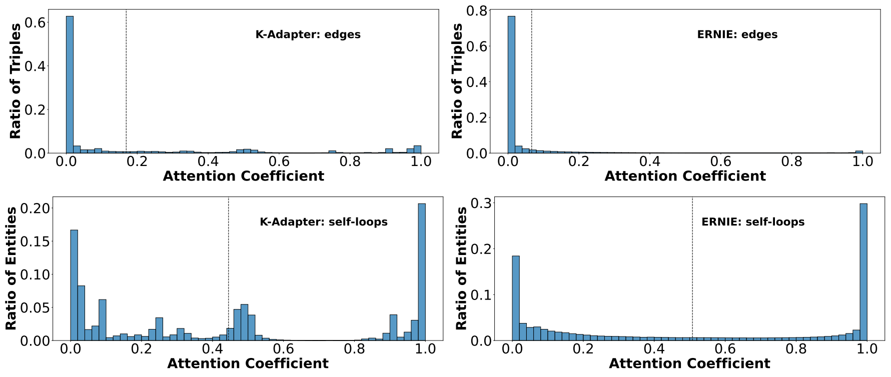


Figure 11: The attention coefficient distributions of edges and self-loops for K-Adapter and ERNIE. The histogram shows the empirical distributions (i.e., frequency), and the blue curves are the Gaussian kernel density estimate. The black dashed vertical lines indicate the average values.

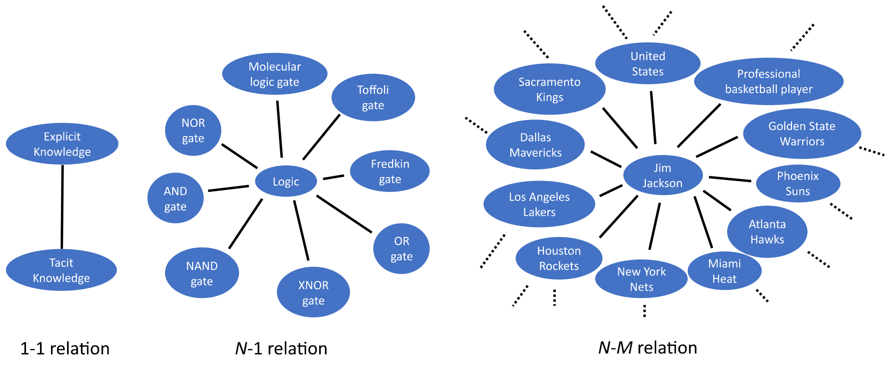


Figure 12: An example of different relations. The black solid lines are relations. We can find that nodes connect to multiple neighbors (e.g., "Logic") are more common than those leaf node (e.g., "OR gate").

Table 8: Analysis of KI interpretation results for K-Adapter and ERNIE in terms of different types of relations (topology feature). The percentages of integrated entities/triples, as well as of CR and CF entities for each type of relations are presented.

Model / Statistics	K-Adapter on T-REx-rc			
	$1-1$ relation	$N-1$ relation	$N-M$ relation	Total
# of triples	21,690	813,674	1,729,644	2,565,008
Integrated triple percentage	58.89%	38.39%	24.00%	28.86%
# of connected entities	21,690	406,837	352,748	781,275
CR entity percentage	41.11%	31.72%	26.02%	29.41%
CF entity percentage	26.40%	30.29%	40.89%	34.97%
Model / Statistics	**ERNIE on Wikidata**			
	$1-1$ relation	$N-1$ relation	$N-M$ relation	Total
# of connected entities	1,799	529,186	2,744,549	3,275,534
Integrated entity percentage	70.65%	42.86%	73.33%	68.39%
CR entity percentage	29.41%	56.07%	26.67%	38.28%
CF entity percentage	23.18%	8.65%	37.10%	32.49%

Table 9: The number of aligned sentences for relations.

Statistics / Relation label	# of triples
Place of birth	134,976
Part of	134,999
Date of death	135,190
Date of birth	135,169
Located in the administrative territorial entity	135,055
Country	135,147
Total	5,565,478