SALKG: Learning From Knowledge Graph Explanations for Commonsense Reasoning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Augmenting pre-trained language models with knowledge graphs (KGs) has achieved success on various commonsense reasoning tasks. However, for a given task instance, the KG, or certain parts of the KG, may not be useful. Although KG-augmented models often use attention to focus on specific KG components, the KG is still always used, and the attention mechanism is never explicitly taught which KG components should be used. Meanwhile, saliency methods [2] can measure how much a KG feature (*e.g.*, graph, node, path) influences the model to make the correct prediction, thus explaining which KG features are useful. This paper explores how saliency explanations can be used to improve KG-augmented models' performance. First, we propose to create coarse (*Is the KG useful?*) and fine (*Which nodes/paths in the KG are useful?*) saliency explanations. Second, to motivate saliency-based supervision, we analyze oracle KG-augmented models which directly use saliency explanations as extra inputs for guiding their attention. Third, we propose SALKG, a framework for KG-augmented models to learn from coarse and/or fine saliency explanations. Given saliency explanations created from a task's training set, SALKG jointly trains the model to predict the explanations, then solve the task by attending to KG features highlighted by the predicted explanations. On three popular commonsense QA benchmarks (CSQA, OBQA, CODAH), we show that SALKG models can yield large performance gains — up to 3.27% on CSQA. [1]

1 Introduction

Natural language processing (NLP) systems generally need common sense to function well in the real world [14]. However, NLP tasks do not always provide the requisite commonsense knowledge as input. Moreover, commonsense knowledge is seldom stated in natural language, making it hard for pre-trained language models (PLMs) [10, 29] — *i.e.*, text encoders — to learn common sense from corpora alone [8, 32]. In contrast to corpora, a knowledge graph (KG) is a rich, structured source of commonsense knowledge, containing numerous facts of the form `(concept1, relation, concept2)`. As a result, many methods follow the ***KG-augmented model*** paradigm, which augments a text encoder with a graph encoder that reasons over the KG (Fig. 2). KG-augmented models have outperformed text encoders on various commonsense reasoning (CSR) tasks, like question answering (QA) (Fig. 1) [25, 4, 30, 51], natural language inference (NLI) [6, 48], and text generation [27, 53].

Since KGs do not have perfect knowledge coverage, they may not contain useful knowledge for all task instances (*e.g.*, if the KG in Fig. 1 only consisted of the gray nodes). Also, even if the KG is useful overall for a given task instance, only some parts of the KG may be useful (*e.g.*, the green nodes in Fig. 1). Ideally, a KG-augmented model would know both if the KG is useful and which parts of the KG are useful. Existing KG-augmented models always assume the KG should be used, but do often use attention [46] to focus on specific KG components (*e.g.*, nodes [12, 41, 50],

[1] Code will be released on GitHub.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

paths [47, 40, 4]) during prediction. Still, the attention mechanism is supervised (end-to-end) only by the task loss, so the model is never *explicitly* taught which KG components should be used. Without component-level supervision, the attention mechanism is more likely to overfit to spurious patterns.

How can we better teach the model whether each KG feature (*e.g.*, graph, node, path) is useful for solving the given task instance? Using the task's ground truth labels, *saliency methods* [2] can score each KG feature's influence on the model making the correct prediction. Whereas attention weights show which KG features the model already used, saliency scores indicate which KG features the model should use. By binarizing these scores, we are able to produce saliency explanations, which can serve as simple targets for training the model's attention mechanism. For example, Fig. 1 shows saliency explanations [`market`=1, `produce`=1, `trading`=0, `merchant`=1, `store`=0, `shop`=0], stating that `market`, `produce`, and `merchant` are useful nodes for answering the question.

In this paper, we investigate how saliency explanations can be used to improve KG-augmented models' performance. First, we propose to create *coarse* (graph-level) and *fine* (node-/path-level) saliency explanations. Since KGs have features at different granularities, saliency explanations can supply a broad range of signals for learning to focus on useful KG features. To create coarse explanations, we introduce an ensemble-based saliency method which measures the performance difference between a KG-augmented model and its corresponding non-KG-augmented model. To create fine explanations, we can adapt any off-the-shelf saliency method, *e.g.*, gradient-based [9] or occlusion-based [24]. Second, to demonstrate the potential of saliency-based supervision, we analyze the performance of *oracle* KG-augmented models, whose attention weights are directly masked with coarse and/or fine saliency explanations.

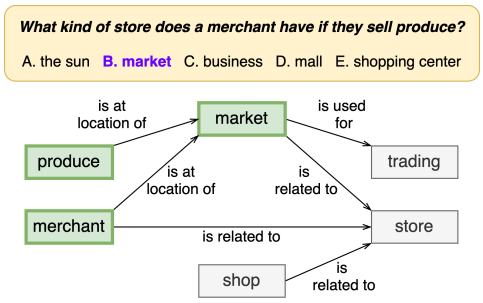

Figure 1: **KG Saliency Explanations for Commonsense QA.** Across different questions, the KG's usefulness can vary considerably. *Coarse* explanations indicate if the KG is useful overall, while *fine* explanations highlight useful nodes or paths. Here, the fine explanations state that the `market`, `produce`, and `merchant` nodes are useful, while the other nodes are not.

Third, as motivated by our oracle model analysis, we propose the *Learning from **Sal**iency Explanations of **KG**-Augmented Models* (**SALKG**) framework. Given coarse and/or fine explanations created from thse task's training set, SALKG jointly trains the model to predict the explanations, then solve the task by attending to KG features highlighted in the predicted explanations. Using saliency explanations to regularize the attention mechanism can help the model generalize better to unseen instances, especially when coarse and fine explanations are used together as complementary learning signals. Indeed, on three standard commonsense QA benchmarks (CSQA, OBQA, CODAH), we show that SALKG models can achieve significant performance gains.

2 Preliminaries

Since KGs abundantly provide structured commonsense knowledge, KG-augmented models are often helpful for solving CSR tasks. CSR tasks are generally formulated as multi-choice QA (discriminative) tasks [44, 33, 20], but sometimes framed as open-ended response (generative) [27, 26] tasks. Given that multi-choice QA has been more extensively studied, we consider CSR in terms of multi-choice QA. Here, we present the multi-choice QA problem setting (Fig. 1) and the structure of KG-augmented models (Fig. 2).

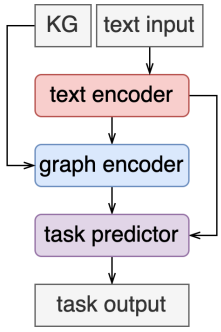

Figure 2: **KG-Augmented Models** fuse knowledge from text and KG inputs to solve CSR tasks.

Problem Definition Given a question q and set of answer choices $A = \{a_i\}$, a multi-choice QA model aims to predict a plausibility score $\rho(q, a_i)$ for each (q, a_i) pair, so that the predicted answer $\hat{a} = \arg\max_{a_i \in A} \rho(q, a_i)$ matches the target answer a^*. Let $q \oplus a_i$ be the text statement formed from (q, a_i), where $\oplus$ denotes concatenation. For example, in Fig. 1, the text statement for $q \oplus a^*$ would be: *What kind of store does a merchant have if they sell produce? market*. We abbreviate $q \oplus a_i$ as x_i and its plausibility score as $\rho(x_i)$.

KG-Augmented Models KG-augmented models use additional supervision from knowledge graphs to solve the multi-choice QA task. They encode the text and KG inputs individually as embeddings, then fuse the two embeddings together to use for prediction. A KG is denoted as $\tilde{\mathcal{G}} = (\tilde{\mathcal{V}}, \tilde{\mathcal{R}}, \tilde{\mathcal{E}})$,

where $\tilde{\mathcal{V}}$, $\tilde{\mathcal{R}}$, and $\tilde{\mathcal{E}}$ are the KG's nodes (concepts), relations, and edges (facts), respectively. An ***edge*** is a directed triple of the form $e = (c_1, r, c_2) \in \tilde{\mathcal{E}}$, in which $c_1, c_2 \in \tilde{\mathcal{V}}$ are ***nodes***, and $r \in \tilde{\mathcal{R}}$ is the ***relation*** between c_1 and c_2. A ***path*** is a connected sequence of edges in the KG. When answering a question, the model does not use the entire KG, since most information in $\tilde{\mathcal{G}}$ will be irrelevant to x_i. Instead, the model uses a smaller, contextualized KG $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{R}_i, \mathcal{E}_i)$, which is built from $\tilde{\mathcal{G}}$ using x_i. $\mathcal{G}_i$ can be constructed heuristically by extracting edges from $\tilde{\mathcal{G}}$ [25, 31], generating edges with a PLM [4], or both [47, 50]. If x_i and $\mathcal{G}_i$ are not discussed in the context of other answer choices, then we further simplify x_i's and $\mathcal{G}_i$'s notation as x and $\mathcal{G}$, respectively. Since the model never uses the *full* KG at once, we use "KG" to refer to $\mathcal{G}$ in the rest of the paper.

As in prior works [25, 4], a KG-augmented model $\mathcal{F}_{\text{KG}}$ has three main components: ***text encoder*** f_{text}, ***graph encoder*** f_{graph}, and ***task predictor*** f_{task} (Fig. 2). Meanwhile, its corresponding non-KG-augmented model $\mathcal{F}_{\text{No-KG}}$ has no graph encoder but has a slightly different task predictor $\bar{f}_{\text{task}}$ which only takes $\mathbf{x}$ as input. In both $\mathcal{F}_{\text{KG}}$ and $\mathcal{F}_{\text{No-KG}}$, the task predictor outputs $\rho(x)$. Let $\mathbf{x}$ and $\mathbf{g}$ be the embeddings of x and $\mathcal{G}$, respectively. Then, the workflows of $\mathcal{F}_{\text{KG}}$ and $\mathcal{F}_{\text{No-KG}}$ are defined below:

$$\mathbf{x} = f_{\text{text}}(x); \quad \mathbf{g} = f_{\text{graph}}(\mathcal{G}, \mathbf{x}); \quad \mathcal{F}_{\text{KG}}(x, \mathcal{G}) = f_{\text{task}}(\mathbf{x} \oplus \mathbf{g}); \quad \mathcal{F}_{\text{No-KG}}(x) = \bar{f}_{\text{task}}(\mathbf{x}).$$

Typically, f_{text} is a PLM [10, 29], f_{graph} is a graph neural network (GNN) [12, 41] or edge/path aggregation model [25, 4, 40], and f_{task} and $\bar{f}_{\text{task}}$ are multilayer perceptrons (MLPs). In general, f_{graph} reasons over $\mathcal{G}$ by encoding either nodes or paths, then using soft attention to pool the encoded nodes/paths into $\mathbf{g}$. Let $\mathcal{L}_{\text{task}}$ be the task loss for training $\mathcal{F}_{\text{KG}}$ and $\mathcal{F}_{\text{No-KG}}$. For multi-choice QA, $\mathcal{L}_{\text{task}}$ is cross-entropy loss, with respect to the distribution over A.

3 Creating KG Saliency Explanations

Now, we show how to create coarse and fine saliency explanations, which tell us if the KG or certain parts of the KG are useful. These explanations can be used as extra inputs for oracle models to mask attention (Sec. 4) or as extra supervision for SALKG models to regularize attention (Sec. 5). We first abstractly define a ***unit*** as either $\mathcal{G}$ itself or a component of $\mathcal{G}$. A unit can be a graph, node, path, *etc.*, and we categorize units as ***coarse*** (the entire graph $\mathcal{G}$) or ***fine*** (a node or path within $\mathcal{G}$) (Table 1). Given a model and task instance $(x, \mathcal{G})$, we define an ***explanation*** as a *binary* indicator of whether a unit u of $\mathcal{G}$ is useful for the model's prediction on $(x, \mathcal{G})$. If u is useful, then u should strongly influence the model to solve the instance correctly. By making explanations binary, we can easily use explanations as masks or learning targets (*i.e.*, binary labels are easier to predict than real-valued scores) for attention weights.

Explanation Setting	Unit
Coarse	KG
Fine (MHGRN)	Node
Fine (PathGen)	Path
Fine (RN)	Path

Table 1: **KG unit types** used for different explanation modes (Sec. 3) and graph encoders (Sec. 4.2).

3.1 Coarse Saliency Explanations

Since $\mathcal{G}$ may not always be useful, a KG-augmented model should ideally know when to use $\mathcal{G}$. Here, the unit u is the graph $\mathcal{G}$. Given instance $(x, \mathcal{G})$, a coarse saliency explanation $y_{\text{c}}(x, \mathcal{G}) \in \{0, 1\}$ indicates if $\mathcal{G}$ helps the model solve the instance. By default, $\mathcal{F}_{\text{KG}}$ assumes $\mathcal{G}$ is used, so we propose an ensemble-based saliency formulation for $y_{\text{c}}(x, \mathcal{G})$. That is, we define $y_{\text{c}}(x, \mathcal{G})$ as stating if $\mathcal{F}_{\text{KG}}$ (*i.e.*, uses $\mathcal{G}$) or $\mathcal{F}_{\text{No-KG}}$ (*i.e.*, does not use $\mathcal{G}$) should be used to solve $(x, \mathcal{G})$. Under this formulation, each $(x, \mathcal{G})$ has coarse units $\mathcal{G}$ and `None`, where `None` means "$\mathcal{G}$ is not used".

To get $y_{\text{c}}(x, \mathcal{G})$, we begin by computing coarse saliency score $s_{\text{c}}(x, \mathcal{G}) \in \mathbb{R}$, which we define as the performance difference between $\mathcal{F}_{\text{KG}}$ and $\mathcal{F}_{\text{No-KG}}$. For QA input $x_i = q \oplus a_i$ and its KG $\mathcal{G}_i$,

$$s_{\text{c}}(x_i, \mathcal{G}_i) = \begin{cases} p_{\text{KG}}(x_i, \mathcal{G}_i) - p_{\text{No-KG}}(x_i), & a_i = a^*, \\ p_{\text{No-KG}}(x_i) - p_{\text{KG}}(x_i, \mathcal{G}_i), & a_i \neq a^*. \end{cases} \quad (1)$$

let $p_{\text{KG}}(x_i, \mathcal{G}_i)$ and $p_{\text{No-KG}}(x_i)$ be the confidence probabilities for x_i predicted by $\mathcal{F}_{\text{KG}}$ and $\mathcal{F}_{\text{No-KG}}$, respectively. Ideally, a QA model should predict higher probabilities for answer choices a_i that are correct, and vice versa. To capture this notion, we define $s_{\text{c}}(x_i, \mathcal{G}_i)$ in Eq. 1, where a^* denotes the correct answer. Note that $s_{\text{c}}(x_i, \mathcal{G}_i)$ is positive if $p_{\text{KG}}(x_i, \mathcal{G}_i)$ is higher than $p_{\text{No-KG}}(x_i)$ for correct choices and lower for incorrect choices. We obtain $y_{\text{c}}(x_i, \mathcal{G}_i)$ by binarizing $s_{\text{c}}(x_i, \mathcal{G}_i)$ to 0 or 1 based on whether it is greater than or less than a threshold T, respectively. We note that the KG is useful if $y_{\text{c}}(x_i, \mathcal{G}_i) = 1$. See the appendix for more details about why we use ensemble-based saliency for coarse explanations and how we tune T.

3.2 Fine Saliency Explanations

Even if $\mathcal{G}$ is useful, not every part of $\mathcal{G}$ may be useful. Hence, fine saliency explanations can define which parts of a KG are actually useful. For a given instance $(x, \mathcal{G})$, we denote the fine saliency explanation for a fine unit u in $\mathcal{G}$ as $y_{\text{f}}(u; x, \mathcal{G}) \in \{0, 1\}$. Fine units can be nodes, paths, *etc.* in the KG. If a graph encoder f_{graph} encodes a certain type of unit, it is natural to define $y_{\text{f}}(u; x, \mathcal{G})$ with respect to such units. For example, MHGRN [12] encodes $\mathcal{G}$'s nodes, so we define MHGRN's fine saliency explanations with respect to nodes. Similar to coarse saliency explanations, to obtain $y_{\text{f}}(u; x, \mathcal{G})$, we first compute fine saliency score $s_{\text{f}}(u; x, \mathcal{G}) \in \mathbb{R}$ and then binarize it. For a QA input $x_i = q \oplus a_i$ and its KG $\mathcal{G}_i$, let u_{ij} be the jth fine unit in $\mathcal{G}_i$ and $p_{\text{KG}}(x_i, \mathcal{G}_i)$ denote the $\mathcal{F}_{\text{KG}}$'s predicted probability for x_i. There are many existing saliency computation method which calculate the importance score of an input for a given label [9, 43, 24]. While $s_f(u_{ij}; x_i, \mathcal{G}_i)$ can be computed via any saliency method, we use gradient-based and occlusion-based methods, since they are good representatives of saliency methods [2].

Let $\phi(u_{ij}; x_i, \mathcal{G}_i)$ denote the raw saliency score given by some saliency method. Gradient-based methods measure an input's saliency by taking the gradient of the model's prediction with respect to the input. We use the *gradient×input* (Grad) method [9], where $\phi(u_{ij}; x_i, \mathcal{G}_i)$ is the dot product of u_{ij}'s embedding and the gradients of $p_{\text{KG}}(x_i, \mathcal{G}_i)$ with respect to u_{ij}. Occlusion-based methods measure an input's saliency as how the model's prediction is affected by erasing that input. We use the *leave-one-out* (Occl) method [24], where $\phi(u_{ij}; x_i, \mathcal{G}_i)$ is the decrease in $p_{\text{KG}}(x_i, \mathcal{G}_i)$ if u_{ij} is removed from $\mathcal{G}_i$, *i.e.*, $\phi(u_{ij}; x_i, \mathcal{G}_i)$ = $p_{\text{KG}}(x_i, \mathcal{G}_i)$ - $p_{\text{KG}}(x_i, \mathcal{G}_i \setminus u_{ij})$.

Intuitively, a unit is more useful if it increases the probability of correct answer choice a^*, and vice versa. Thus, we define the saliency scores for a unit u_{ij} as Eq. 2. Next, we binarize these scores to get $y_{\text{f}}(u_{ij}; x_i, \mathcal{G}_i)$ by selecting the top-k%-scoring units in $\mathcal{G}_i$ and setting $y_{\text{f}}(u_{ij}; x_i, \mathcal{G}_i) = 1$ (*i.e.*, u_{ij} is useful). For all other units in $\mathcal{G}$, we set $y_{\text{f}}(u_{ij}; x_i, \mathcal{G}_i) = 0$ (*i.e.*, u_{ij} is not useful). See the appendix for more details about fine saliency methods and tuning k.

$$s_{\text{f}}(u_{ij}; x_i, \mathcal{G}_i) = \begin{cases} \phi(u_{ij}; x_i, \mathcal{G}_i), & a_i = a^* \\ -\phi(u_{ij}; x_i, \mathcal{G}_i), & a_i \neq a^* \end{cases} \quad (2)$$

4 ORACLE: Using KG Saliency Explanations as Inputs

In this section, we analyze saliency explanations' potential to improve KG-augmented models' performance. Note that creating saliency explanations requires the task's ground truth labels (Sec. 3), so directly using explanations at test time is infeasible. Still, before exploring ways to leverage training set explanations at inference time, we first establish upper bounds on how much models can benefit from saliency explanations. Here, we study three key questions: **(1)** *Does the model improve when provided oracle access to coarse/fine explanations?* **(2)** *Are coarse and fine explanations complementary?* **(3)** *How do gradient-based explanations compare to occlusion-based explanations?*

4.1 ORACLE Models

ORACLE models are KG-augmented models with oracle access to saliency explanations. An ORACLE model uses ground truth labels to create explanations (even during inference time), and then uses the explanations as extra inputs to perform hard attention over the units. We define the model attention weights that are modified based on saliency explanations as ***saliency weights***. Below, we introduce the ORACLE-Coarse, ORACLE-Fine, and ORACLE-Hybrid models, shown in Fig. 3a-c.

ORACLE-Coarse ORACLE-Coarse ($\mathcal{F}_{\text{c}}^*$) uses coarse explanations to do hard attention over $\mathcal{F}_{\text{KG}}$'s and $\mathcal{F}_{\text{No-KG}}$'s predictions. First, $\mathcal{F}_{\text{KG}}$ and $\mathcal{F}_{\text{No-KG}}$ are trained separately, then frozen. Next, for each instance $(x, \mathcal{G})$, they are used to create a coarse explanation $y_{\text{c}}(x, \mathcal{G}) \in \{0, 1\}$. Then, $\mathcal{F}_{\text{c}}^*$ is defined as an ensemble model that performs hard attention over coarse units ($\mathcal{G}$ and `None`) by weighting $\mathcal{F}_{\text{KG}}$'s prediction with $y_{\text{c}}(x, \mathcal{G})$ and $\mathcal{F}_{\text{No-KG}}$'s prediction with $1 - y_{\text{c}}(x, \mathcal{G})$ (Table 2; Fig. 3a). In other words, $y_{\text{c}}(x, \mathcal{G})$ and $1 - y_{\text{c}}(x, \mathcal{G})$ are the saliency weights for $\mathcal{F}_{\text{c}}^*$.

ORACLE-Fine ORACLE-Fine ($\mathcal{F}_{\text{f}}^*$) has the same architecture as $\mathcal{F}_{\text{KG}}$ and uses fine explanations to do hard attention over fine units (*i.e.*, nodes or paths in $\mathcal{G}$). First, $\mathcal{F}_{\text{KG}}$ is trained, then frozen. As usual, $\mathcal{F}_{\text{KG}}$ uses soft attention over fine units in $\mathcal{G}$ to compute graph embedding $\mathbf{g}$ (Sec. 2). Then, for each fine unit u in $\mathcal{G}$, $\mathcal{F}_{\text{KG}}$ is used to create fine explanation $y_{\text{f}}(u; x, \mathcal{G}) \in \{0, 1\}$. Let $\hat{y}_{\text{f}}(u; x, \mathcal{G}) \in [0, 1]$ denote $\mathcal{F}_{\text{f}}^*$'s soft attention weight for u. We train $\mathcal{F}_{\text{f}}^*$ the same way as $\mathcal{F}_{\text{KG}}$, except

Model	Output	Saliency Weights
ORACLE-Coarse	$\mathcal{F}_{\text{c}}^*(x,\mathcal{G}) = y_{\text{c}}(x,\mathcal{G})\mathcal{F}_{\text{KG}}(x,\mathcal{G}) + (1-y_{\text{c}}(x,\mathcal{G}))\mathcal{F}_{\text{No-KG}}(x)$	$[y_{\text{c}}(x,\mathcal{G}), 1-y_{\text{c}}(x,\mathcal{G})]$
ORACLE-Fine	$\mathcal{F}_{\text{f}}^*(x,\mathcal{G}) \sim \mathcal{F}_{\text{KG}}(x,\mathcal{G})$	$\hat{y}_{\text{f}}(x,\mathcal{G}) \odot y_{\text{f}}(x,\mathcal{G})$
ORACLE-Hybrid	$\mathcal{F}_{\text{h}}^*(x,\mathcal{G}) = y_{\text{h}}(x,\mathcal{G})\mathcal{F}_{\text{f}}^*(x,\mathcal{G}) + (1-y_{\text{h}}(x,\mathcal{G}))\mathcal{F}_{\text{No-KG}}(x)$	$[y_{\text{h}}(x,\mathcal{G}), 1-y_{\text{h}}(x,\mathcal{G})]$

Table 2: **Comparison of ORACLE Models.** For each ORACLE Model, we show its output and saliency weights. Note that the explanations are given (not predicted), so there is no $\mathcal{L}_{\text{sal}}$. While $\mathcal{F}_{\text{c}}^*$ and $\mathcal{F}_{\text{h}}^*$ are both ensembles of $\mathcal{F}_{\text{KG}}$ and $\mathcal{F}_{\text{No-KG}}$, $\mathcal{F}_{\text{f}}^*$ has the same architecture as $\mathcal{F}_{\text{KG}}$ (denoted by $\sim$) besides the attention masking.

	CSQA Test Accuracy (%)						OBQA Test Accuracy (%)					
	MHGRN		PathGen		RN		MHGRN		PathGen		RN	
Model	BERT	RoBERTa	BERT	RoBERTa	BERT	RoBERTa	BERT	RoBERTa	BERT	RoBERTa	BERT	RoBERTa
No-KG	55.44	70.59	55.44	70.59	55.44	70.59	53.60	68.40	53.60	68.40	53.60	68.40
KG	56.57	73.33	56.65	72.04	55.60	71.07	53.20	69.80	55.00	67.80	58.60	70.20
No-KG + KG	56.57	71.39	57.45	73.00	56.73	68.49	55.60	70.60	54.40	70.6	53.40	69.60
ORACLE-Coarse	66.16	81.39	68.57	80.10	67.28	79.69	70.60	79.40	65.00	76.60	69.00	79.00
ORACLE-Fine (Grad)	74.86	76.15	79.61	87.35	81.39	83.24	67.60	72.60	73.80	73.40	68.00	62.80
ORACLE-Fine (Occl)	91.06	87.99	79.61	75.34	73.73	68.41	77.00	71.20	83.60	62.60	55.60	61.40
ORACLE-Hybrid (Grad)	85.50	84.21	**90.49**	92.83	**92.26**	93.56	80.80	84.80	85.60	**92.80**	**85.40**	**86.80**
ORACLE-Hybrid (Occl)	**95.89**	**98.63**	88.96	**96.78**	85.25	**95.25**	**87.00**	**89.60**	**92.80**	90.60	67.40	80.60

Table 3: **ORACLE Performance on CSQA and OBQA**

each $\hat{y}_{\text{f}}(u;x,\mathcal{G})$ is masked with $y_{\text{f}}(u;x,\mathcal{G})$, *i.e.*, $\hat{y}_{\text{f}}(u;x,\mathcal{G}) \leftarrow \hat{y}_{\text{f}}(u;x,\mathcal{G}) \odot y_{\text{f}}(u;x,\mathcal{G})$, where $\odot$ denotes the elementwise multiplication (Table 2; Fig. 3b). This means only units with $y_{\text{f}}(u;x,\mathcal{G}) = 1$ will have $\hat{y}_{\text{f}}(u;x,\mathcal{G}) > 0$ and thus be able to influence $\mathcal{F}_{\text{f}}^*$'s prediction. Let $y_{\text{f}}(x,\mathcal{G})$ denote the fine saliency explanations for all units in the graph (similarly, $\hat{y}_{\text{f}}(x,\mathcal{G})$). Then, $\hat{y}_{\text{f}}(x,\mathcal{G}) \odot y_{\text{f}}(x,\mathcal{G})$ are the saliency weights for $\mathcal{F}_{\text{f}}^*$.

ORACLE-Hybrid ORACLE-Hybrid ($\mathcal{F}_{\text{h}}^*$) unifies ORACLE-Coarse and ORACLE-Fine as a single model, thus leveraging the hierarchy inherent in KG saliency explanations. First, $\mathcal{F}_{\text{f}}^*$ (which uses fine explanations) and $\mathcal{F}_{\text{No-KG}}$ are separately trained, then frozen. Then, for each $(x,\mathcal{G})$, $\mathcal{F}_{\text{f}}^*$ and $\mathcal{F}_{\text{No-KG}}$ are used to create $y_{\text{h}}(x,\mathcal{G}) \in \{0,1\}$, which we define as the coarse explanation for $\mathcal{F}_{\text{f}}^*$ and $\mathcal{F}_{\text{No-KG}}$. $y_{\text{h}}(x,\mathcal{G})$ is computed the same way as $y_{\text{c}}(x,\mathcal{G})$, besides replacing $\mathcal{F}_{\text{KG}}$ with $\mathcal{F}_{\text{f}}^*$. Finally, similar to $\mathcal{F}_{\text{c}}^*$, $\mathcal{F}_{\text{h}}^*$ is an ensemble that performs hard attention over coarse units by weighting $\mathcal{F}_{\text{f}}^*$'s prediction with $y_{\text{h}}(x,\mathcal{G})$ and $\mathcal{F}_{\text{No-KG}}$'s prediction with $1 - y_{\text{h}}(x,\mathcal{G})$ (Table 2; Fig. 3c). That is, $y_{\text{h}}(x,\mathcal{G})$ and $1 - y_{\text{h}}(x,\mathcal{G})$ are the saliency weights for $\mathcal{F}_{\text{h}}^*$.

4.2 Evaluation Protocol

We use the CSQA [44] and OBQA [33] multi-choice QA datasets. For CSQA, we use the accepted in-house data split from [25], as official test labels are not public. As in prior works, we use the ConceptNet [42] KG for both datasets. We report accuracy, the standard metric for multi-choice QA. For $\mathcal{F}_{\text{No-KG}}$ and $\mathcal{F}_{\text{KG}}$, we pick the best model over three seeds, then use them to create explanations for ORACLE models. We use thresholds $T = 0.01$ and $k = 10$ for coarse and fine explanations, respectively. For text encoders, we use BERT(-Base) [10] and RoBERTa(-Large) [29]. For graph encoders, we use MHGRN [12], PathGen [47], and Relation Network (RN) [40, 25]. MHGRN has node units, while PathGen and RN have path units. As ***baseline models***, we use $\mathcal{F}_{\text{No-KG}}$, $\mathcal{F}_{\text{KG}}$, and $\mathcal{F}_{\text{No-KG}} + \mathcal{F}_{\text{KG}}$, where $\mathcal{F}_{\text{No-KG}} + \mathcal{F}_{\text{KG}}$ is an ensemble whose prediction is the mean of $\mathcal{F}_{\text{No-KG}}$'s and $\mathcal{F}_{\text{KG}}$'s predictions. ORACLE and baseline models are trained only with task loss $\mathcal{L}_{\text{task}}$.

4.3 Analysis

In Table 3, we show CSQA and OBQA performance for the baseline and ORACLE models. We analyze these results via the three questions below.

Does the model improve when provided oracle access to coarse/fine explanations? Yes. ORACLE-Coarse beats all baselines, while ORACLE-Fine beats all baselines except on OBQA RN+RoBERTa. Our ORACLE pilot results motivate us to develop a framework for models to improve performance by learning from coarse/fine explanations. Also, on average, ORACLE-Fine outperforms ORACLE-Coarse, which suggests that fine explanations may often provide richer signal than their coarse counterparts. Indeed, fine explanations indicate the saliency of every unit in the KG, while coarse explanations only indicate the saliency of the KG as a whole.

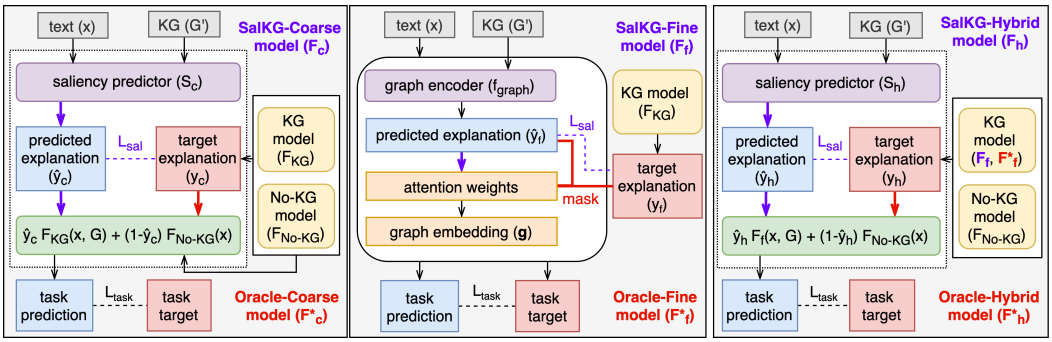

Figure 3: **Schematics for ORACLE and SALKG Models.** Red arrows indicate the ORACLE pipeline, where the target explanation is provided as input. Blue arrows indicate the SALKG pipeline, where the target explanation is used as supervision for the predicted explanation. In SALKG-Coarse and SALKG-Hybrid, the saliency predictor has the same architecture as $\mathcal{F}_{\text{KG}}$. Meanwhile, ORACLE-Fine and SALKG-Fine (shown as white module, with text encoder and task predictor omitted) both have the same architecture as $\mathcal{F}_{\text{KG}}$.

Are coarse and fine explanations complementary? Yes. Across all settings, ORACLE-Hybrid performs significantly better than ORACLE-Coarse and ORACLE-Fine. This suggests that coarse and fine explanations are complementary and that it is effective to leverage both hierarchically.

How do gradient-based explanations compare to occlusion-based explanations? Overall, Grad and Occl perform similarly. Grad performs better on some settings (*e.g.*, MHGRN), while Occl performs better on others (*e.g.*, RN). See the appendix and Table 7 for more Grad *vs.* Occl experiments.

In our analysis, KG-augmented models achieve large performance gains when provided explanations as input. This suggests that, if oracle explanations can somehow be *predicted* accurately during inference without using the ground truth labels, then KG-augmented models can still achieve performance gains without directly using explanations as input. This motivates us to train KG-augmented models with explanation-based supervision via SALKG, which we define next.

5 SALKG: Using KG Saliency Explanations as Supervision

Based on the analysis from Sec. 4.3, we propose the SALKG framework for KG-augmented models to learn from coarse/fine saliency explanations. Whereas ORACLE models (Sec. 4.1) use explanations directly as extra inputs, SALKG models only use them as extra supervision during training phase. With explanations created from the training set via $\mathcal{F}_{\text{KG}}$ and $\mathcal{F}_{\text{No-KG}}$, SALKG models are jointly trained to predict the explanations (via saliency loss $\mathcal{L}_{\text{sal}}$) and use the predicted explanations to solve the task (via task loss $\mathcal{L}_{\text{task}}$). Thus, SALKG models have the following objective: $\mathcal{L}_{\text{S}} = \mathcal{L}_{\text{task}} + \lambda \mathcal{L}_{\text{sal}}$, where $\lambda \geq 0$ is a loss weighting parameter. This multitask objective not only encourages SALKG models to focus on useful KG units for solving the task, but also to learn more general graph/node/path representations. Below, we present SALKG-Coarse, SALKG-Fine, and SALKG-Hybrid models.

SALKG-Coarse Unlike ORACLE-Coarse, SALKG-Coarse ($\mathcal{F}_{\text{c}}$) is not given oracle coarse explanation $y_{\text{c}}(x, \mathcal{G})$ as input. Instead, a saliency predictor $\mathcal{S}_{\text{c}}$ (with the same architecture as $\mathcal{F}_{\text{KG}}$) is trained to predict the oracle coarse explanation. $\mathcal{S}_{\text{c}}$ predicts coarse explanation as probability $\hat{y}_{\text{c}}(x, \mathcal{G}) \in [0, 1]$. $\mathcal{F}_{\text{c}}$'s output is an ensemble that does soft attention over coarse units by weighting $\mathcal{F}_{\text{KG}}$'s and $\mathcal{F}_{\text{No-KG}}$'s predictions with saliency weights $\hat{y}_{\text{c}}(x, \mathcal{G})$ and $1 - \hat{y}_{\text{c}}(x, \mathcal{G})$, respectively (Table 4; Fig. 3a). Here, $\mathcal{L}_{\text{sal}}(\hat{y}_{\text{c}}(x, \mathcal{G}), y_{\text{c}}(x, \mathcal{G}))$ is the cross-entropy loss.

SALKG-Fine Similarly, SALKG-Fine ($\mathcal{F}_{\text{f}}$) is not given oracle fine explanation $y_{\text{f}}(u; x, \mathcal{G})$ as input, although both have the same architecture as $\mathcal{F}_{\text{KG}}$. Instead, for each fine unit u, $\mathcal{F}_{\text{f}}$'s attention mechanism is trained to predict $y_{\text{f}}(u; x, \mathcal{G})$ as soft attention weight $\hat{y}_{\text{f}}(u; x, \mathcal{G}) \in [0, 1]$ (Table 4; Fig. 3b). As before, $\hat{y}_{\text{f}}(x, \mathcal{G}) = [\hat{y}_{\text{f}}(u; x, \mathcal{G})]_{u \in \mathcal{G}}$ are the soft attention weights for $(x, \mathcal{G})$, while $y_{\text{f}}(x, \mathcal{G}) = [y_{\text{f}}(u; x, \mathcal{G})]_{u \in \mathcal{G}}$ are the fine explanations for $(x, \mathcal{G})$. Then, $\hat{y}_{\text{f}}(x, \mathcal{G})$ are the saliency weights for $\mathcal{F}_{\text{f}}$, trained with KL divergence loss $\mathcal{L}_{\text{sal}}(\hat{y}_{\text{f}}(x, \mathcal{G}), y_{\text{f}}(x, \mathcal{G}))$.

SALKG-Hybrid Similar to the other SALKG variants, SALKG-Hybrid ($\mathcal{F}_{\text{h}}$) does not use any oracle explanations. Like in SALKG-Coarse, a saliency predictor $\mathcal{S}_{\text{h}}$ is trained to predict oracle coarse explanation $y_h(x, \mathcal{G})$ (Sec. 4.1). Predicted coarse explanation probabilities $\hat{y}_{\text{h}}(x, \mathcal{G}) \in [0, 1]$ are then used as soft attention over coarse units by weighting $\mathcal{F}_{\text{f}}$'s and $\mathcal{F}_{\text{No-KG}}$'s predictions with

Model	Output	Saliency Weights	Saliency Loss ($\mathcal{L}_{\text{sal}}$)
SALKG-Coarse	$\mathcal{F}_{\text{c}}(x,\mathcal{G}) = \hat{y}_{\text{c}}(x,\mathcal{G})\,\mathcal{F}_{\text{KG}}(x,\mathcal{G}) + (1-\hat{y}_{\text{c}}(x,\mathcal{G}))\,\mathcal{F}_{\text{No-KG}}(x)$	$[\hat{y}_{\text{c}}(x,\mathcal{G}), 1-\hat{y}_{\text{c}}(x,\mathcal{G})]$	$\text{CE}(\hat{y}_{\text{c}}(x,\mathcal{G}), y_{\text{c}}(x,\mathcal{G}))$
SALKG-Fine	$\mathcal{F}_{\text{f}}(x,\mathcal{G}) \sim \mathcal{F}_{\text{KG}}(x,\mathcal{G})$	$\hat{y}_{\text{f}}(x,\mathcal{G})$	$\text{KL}(\hat{y}_{\text{f}}(x,\mathcal{G}), y_{\text{f}}(x,\mathcal{G}))$
SALKG-Hybrid	$\mathcal{F}_{\text{h}}(x,\mathcal{G}) = \hat{y}_{\text{h}}(x,\mathcal{G})\mathcal{F}_{\text{f}}(x,\mathcal{G}) + (1-\hat{y}_{\text{h}}(x,\mathcal{G}))\mathcal{F}_{\text{No-KG}}(x)$	$[\hat{y}_{\text{h}}(x,\mathcal{G}), 1-\hat{y}_{\text{h}}(x,\mathcal{G})]$	$\text{CE}(\hat{y}_{\text{h}}(x,\mathcal{G}), y_{\text{h}}(x,\mathcal{G}))$

Table 4: **Comparison of SALKG Models.** For each SALKG Model, we show its output, saliency weights, and $\mathcal{L}_{\text{sal}}$. While $\mathcal{F}_{\text{c}}$ and $\mathcal{F}_{\text{h}}$ are both ensembles, $\mathcal{F}_{\text{f}}$ has the same architecture as $\mathcal{F}_{\text{KG}}$ (denoted by $\sim$). "CE" denotes cross-entropy loss, while "KL" denotes KL divergence loss.

weights $\hat{y}_{\text{h}}(x,\mathcal{G})$ and $1-\hat{y}_{\text{h}}(x,\mathcal{G})$, respectively (Table 4; Fig. 3c). Here, $\mathcal{L}_{\text{sal}}(\hat{y}_{\text{h}}(x,\mathcal{G}), y_{\text{h}}(x,\mathcal{G}))$ is cross-entropy loss.

6 Experiments

6.1 Evaluation Protocol

We evaluate SALKG models on the CSQA [44], OBQA [33], and CODAH [5] multi-choice QA datasets. We follow the same evaluation protocol in Sec. 4.2, except we now also report mean and standard deviation over three seeds. In addition to the baselines in Sec. 4.2, we consider two new baselines, RANDOM and HEURISTIC, which help show that coarse/fine saliency explanations provide strong learning signal for KG-augmented models to focus on useful KG features.

RANDOM RANDOM is a variant of SALKG where each unit's explanation is random. RANDOM-Coarse is like SALKG-Coarse, but with each $y_{\text{c}}(x,\mathcal{G})$ uniformly sampled from $\{0,1\}$. RANDOM-Fine is like SALKG-Fine, but randomly picking $k\%$ of units in $\mathcal{G}$ to set $y_{\text{f}}(u;x,\mathcal{G}) = 1$. RANDOM-Hybrid is like SALKG-Hybrid, but with each $y_h(x,\mathcal{G})$ uniformly sampled from $\{0,1\}$ as well as using RANDOM-Fine instead of SALKG-Fine.

HEURISTIC Each $\mathcal{G}$ has three node types: question nodes (*i.e.*, nodes in q), answer nodes (*i.e.*, nodes in a_i), and intermediate nodes (*i.e.*, other nodes) [25]. Let QA nodes be nodes in q or a_i. HEURISTIC is a variant of SALKG where each unit's explanation is based on the presence of QA nodes in $\mathcal{G}$. Let $\bar{N}$ be the mean number of QA nodes per KG (in train set), and let $N(\mathcal{G})$ be the number of QA nodes in $\mathcal{G}$. HEURISTIC-Coarse is like SALKG-Coarse, except $y_{\text{c}}(x,\mathcal{G}) = 1$ if and only if $N(\mathcal{G}) > \bar{N}$. HEURISTIC-Fine is like SALKG-Fine, but how $y_{\text{f}}(u;x,\mathcal{G})$ is set depends on whether the fine units are nodes or paths. For node units, $y_{\text{f}}(u;x,\mathcal{G}) = 1$ if and only if u is a QA node. For path units, $y_{\text{f}}(u;x,\mathcal{G}) = 1$ if and only if u consists only of QA nodes. HEURISTIC-Hybrid is like SALKG-Hybrid, but with $y_h(x,\mathcal{G}) = 1$ if and only if $N(\mathcal{G}) > \bar{N}$, while HEURISTIC-Fine is used instead of SALKG-Fine.

6.2 Main Results

Table 5 shows performance on CSQA, while Table 6 shows performance on OBQA and CODAH. Best performance is highlighted in green, second-best performance is highlighted in blue, and best non-SALKG performance is highlighted in red (if it is not already green or blue). For SALKG (unlike ORACLE), we find that Occl usually outperforms Grad, so we only report Occl performance in Tables 5-6. For a comparison of Grad and Occl on SALKG, see Table 7 and the appendix. Being an ensemble, No-KG + KG tends to beat both No-KG and KG if both have similar performance. Otherwise, No-KG + KG performance is in between No-KG's and KG's.

Across all datasets, we find that on average SALKG-Hybrid performs the best overall, while SALKG-Coarse is second-best. On CSQA, SALKG-Hybrid has the highest performance on every setting except RoBERTa+MHGRN. In particular, on RoBERTa+RN and BERT+PathGen, SALKG-Hybrid beats max(No-KG, KG, No-KG + KG) by large margins of 3.27% and 3.06%, respectively. On OBQA, SALKG-Hybrid still improves performance but is not as dominant, beating max(No-KG, KG, No-KG + KG) by 1.07% on RoBERTa+MHGRN. On CODAH, SALKG-Hybrid similarly beats max(No-KG, KG, No-KG + KG) by 1.63% on RoBERTa+MHGRN. Meanwhile, SALKG-Fine is not as consistently strong, but still shows improvement on some settings (e.g., RoBERTa+MHGRN and RoBERTa+RN on CSQA). This shows that learning from explanations is generally effective for improving KG-augmented models' performance, especially when both coarse and fine explanations are used to provide complementary learning signals for SALKG-Hybrid. In the appendix, we compare SALKG to additional baseline KG-augmented models and find that SALKG is competitive.

Model	CSQA Test Accuracy (%)					
	MHGRN		PathGen		RN	
	BERT	RoBERTa	BERT	RoBERTa	BERT	RoBERTa
No-KG	53.13 (±2.34)	69.65 (±1.06)	53.13 (±2.34)	69.65 (±1.06)	53.13 (±2.34)	69.65 (±1.06)
KG	57.48 (±0.89)	73.14 (±0.78)	56.54 (±0.73)	72.58 (±0.57)	56.46 (±1.22)	71.37 (±1.20)
No-KG + KG	56.14 (±2.29)	72.14 (±0.67)	57.29 (±1.30)	71.44 (±0.71)	56.86 (±1.10)	70.53 (±1.82)
RANDOM-Coarse	57.10 (±0.53)	72.28 (±0.58)	56.00 (±0.37)	71.56 (±0.73)	55.63 (±0.96)	71.18 (±0.51)
RANDOM-Fine	44.75 (±0.91)	71.36 (±0.58)	56.54 (±2.02)	69.59 (±3.63)	50.85 (±1.07)	68.79 (±0.17)
RANDOM-Hybrid	52.22 (±1.65)	71.90 (±0.58)	55.41 (±0.57)	71.45 (±0.69)	53.94 (±0.55)	70.32 (±0.95)
HEURISTIC-Coarse	57.05 (±0.28)	71.82 (±0.23)	57.10 (±0.17)	71.77 (±0.05)	55.76 (±0.21)	71.85 (±0.34)
HEURISTIC-Fine	52.78 (±1.34)	71.84 (±0.94)	53.85 (±1.29)	71.26 (±0.81)	50.60 (±0.81)	67.23 (±1.66)
HEURISTIC-Hybrid	56.85 (±0.06)	72.47 (±0.12)	56.19 (±0.20)	71.79 (±0.47)	51.79 (±0.25)	64.97 (±1.26)
SALKG-Coarse	57.78 (±0.35)	74.05 (±0.14)	58.23 (±0.37)	72.79 (±0.12)	57.59 (±0.33)	72.44 (±0.16)
SALKG-Fine	56.78 (±2.14)	73.65 (±0.21)	57.64 (±2.12)	71.39 (±1.54)	56.86 (±0.41)	71.58 (±1.10)
SALKG-Hybrid	59.12 (±0.28)	73.41 (±0.16)	60.35 (±0.32)	73.11 (±1.00)	58.80 (±0.19)	74.64 (±0.09)

Table 5: **SALKG Performance on CSQA**

Model (RoBERTa)	OBQA Test Accuracy (%)			CODAH Test Accuracy (%)	
	MHGRN	PathGen	RN	MHGRN	PathGen
No-KG	68.73 (±0.31)	68.73 (±0.31)	68.73 (±0.31)	83.96 (±0.79)	83.96 (±0.79)
KG	68.87 (±2.16)	68.40 (±1.59)	66.80 (±4.73)	84.02 (±1.27)	84.02 (±1.62)
No-KG + KG	68.53 (±0.95)	69.67 (±1.45)	69.40 (±0.35)	84.08 (±1.46)	84.69 (±1.48)
RANDOM-Coarse	68.80 (±0.35)	69.60 (±0.92)	68.20 (±1.00)	84.62 (±0.55)	86.07 (±0.89)
RANDOM-Fine	60.33 (±5.33)	55.13 (±7.00)	48.53 (±4.82)	84.02 (±0.89)	75.74 (±4.71)
RANDOM-Hybrid	68.33 (±0.40)	69.53 (±0.31)	69.27 (±0.12)	84.80 (±0.10)	84.68 (±0.18)
HEURISTIC-Coarse	68.30 (±1.27)	55.00 (±0.00)	68.27 (±0.23)	85.11 (±0.10)	85.59 (±0.00)
HEURISTIC-Fine	57.27 (±3.76)	51.80 (±2.95)	50.53 (±3.51)	83.72 (±0.92)	82.64 (±2.61)
HEURISTIC-Hybrid	68.47 (±0.23)	68.40 (±0.00)	68.60 (±0.20)	84.80 (±0.10)	85.23 (±0.00)
SALKG-Coarse	69.93 (±0.31)	70.00 (±0.35)	71.47 (±0.58)	85.41 (±0.18)	85.95 (±0.18)
SALKG-Fine	65.07 (±1.70)	51.60 (±1.13)	62.33 (±0.90)	84.08 (±1.14)	83.36 (±0.81)
SALKG-Hybrid	70.73 (±0.12)	70.07 (±0.12)	70.80 (±0.00)	85.71 (±0.10)	84.44 (±0.63)

Table 6: **SALKG Performance on OBQA and CODAH**

Second, across all datasets, we find that SALKG models generally outperform their RANDOM and HEURISTIC counterparts. On CSQA, this trend is very strong, as SALKG beats RANDOM and HEURISTIC on every setting. This is evidence that explanations created from saliency methods can provide better learning signal than those created randomly or from simple heuristics. However, on OBQA and CODAH, SALKG does not always not always beat RANDOM and HEURISTIC, although the top-2 models per column are usually SALKG models. Interestingly, RANDOM sometimes outperforms $\max$(No-KG, KG, No-KG + KG). In particular, for RoBERTa+PathGen on CODAH, RANDOM-Coarse beats all other models, although SALKG-Coarse is almost as good. For RoBERTa+PathGen on CODAH, SALKG's limited performance may stem from the coarse explanations being unable to consistently discriminate between No-KG and KG, which can happen if No-KG and KG have very similar predictions. See the appendix for more analysis on this.

6.3 Ablation Studies

In Table 7, we validate our SALKG design choices with ablation studies. We report dev accuracy for BERT+MHGRN and BERT+PathGen on CSQA.

Are ensemble-based coarse explanations effective? By default, SALKG-Coarse uses our proposed ensemble-based coarse explanations (Sec. 3.1). Alternatively, we consider using Grad and Occl to create coarse explanations. For Grad, we compute ϕ the same way as in Sec. 3.2, except using graph embedding $\mathbf{g}$ instead of node/path embeddings. Since a zero vector would have zero gradient, this is equivalent to comparing $\mathbf{g}$ to a zero vector baseline. For Occl, we compute ϕ as the decrease in p_{KG} if $\mathbf{g}$ is replaced with a zero vector. For both Grad and Occl, we set $s_{\text{c}} = \phi$. In Table 7, we see that our default SALKG-Coarse, significantly outperforms SALKG-Coarse

Model (BERT)	CSQA Dev Accuracy (%)	
	MHGRN	PathGen
SALKG-Coarse	**59.49** (±0.05)	**60.72** (±0.58)
- w/ Grad	56.84 (±2.27)	56.18 (±2.31)
- w/ Occl	57.60 (±0.74)	56.32 (±1.66)
SALKG-Fine (Occl)	**57.28** (±0.95)	**59.13** (±2.35)
- w/ Grad	56.05 (±1.03)	58.80 (±1.08)
SALKG-Hybrid (Occl)	59.92 (±0.31)	**60.88** (±0.05)
- w/ Grad	**60.17** (±0.21)	59.71 (±0.08)
SALKG-Fine (Occl)	**57.28** (±0.95)	**59.13** (±2.35)
- w/ Random Prune	50.61 (±0.68)	54.10 (±2.13)
- w/ Heuristic Prune	50.72 (±0.46)	50.53 (±0.74)
SALKG-Fine (Occl)	**57.28** (±0.95)	**59.13** (±2.35)
- w/ BCE Sal. Loss	50.83 (±1.75)	55.15 (±2.58)

Table 7: **Ablation Studies**

with both Grad and Occl. In the appendix, we further discuss why Grad and Occl are ill-suited for creating coarse explanations.

For* SALKG*, is Occl better than Grad? In Tables 5-6, we report SALKG-Fine and SALKG-Hybrid performance with Occl fine explanations. In Table 7, we compare Occl and Grad on SALKG-Fine and SALKG-Hybrid. Overall, Occl slightly outperforms Grad, although Grad beats Occl on MHGRN for SALKG-Hybrid. Their relative performance could also depend on the choice of top-$k\%$, which we plan to explore later. In the appendix, we further compare Occl and Grad on other settings.

How does* SALKG-Fine*'s soft KG pruning compare to hard KG pruning? SALKG-Fine does soft pruning of unhelpful fine units via attention. We compare SALKG-Fine to two baselines where the KG is filtered via hard pruning, which cannot be part of end-to-end training. For RANDOM Prune and HEURISTIC Prune, we respectively create RANDOM and HEURISTIC explanations, then hard prune all negative units from the KG. The KG-augmented model then uses the pruned KG as its KG input. In Table 7, we see that SALKG-Fine significantly outperforms the two baselines, showing the benefits of jointly training the model on saliency and QA prediction.

Is it effective to train* SALKG-Fine *with KL divergence? We train SALKG-Fine's explanation predictor (*i.e.*, attention mechanism) using KL divergence as the saliency loss. Thus, within a KG, the distribution over attention weights constitutes a single prediction. Alternatively, we could treat each attention weight as a separate prediction and train the attention mechanism using binary cross entropy (BCE) loss. In Table 7, we find that using KL divergence yields much higher performance than using BCE loss. This suggests that the attention weights should not be trained separately, as each attention weight is highly dependent on other attention weights in the same KG.

6.4 Case Studies

We visualize coarse/fine explanations created from BERT+PathGen on CSQA, with 1-hop or 2-hop paths as fine units. For coarse explanations, we show examples of positive (*i.e.*, useful) and negative KGs. Since KGs are too large to show here, we uniformly sample three paths per KG. For the positive KG example, the question is *James loved to play violin. He did it in his spare time because he found it what?*, the answer choice is *relaxing*, and the target answer is *relaxing*. Its paths are: **(1)** `play` –[is related to]–> `x` <–[is used for]– `relaxing` , **(2)** `violin` –[is used for]–> `x` –[is used for]–> `relaxing` , and **(3)** `time` <–[has subevent]– `x` –[has subevent]–> `relax` . For the negative KG example, the question is *Where do soldiers not deployed eat their food?*, the answer choice is *neighbor's house*, and the target answer is *military base*. Its paths are: **(1)** `soldier` <–[is related to]– `x` <–[is related to]– `house` , **(2)** `eat` –[is related to]–> `x` –[is at location of]–> `house` , and **(3)** `food` <–[is related to]– `x` –[is at location of]–> `house` . For fine explanations, we show examples of positive and negative paths from the same KG. Here, the question is *Where can you find a bar before traveling a long distance?*, the answer choice is *airport*, and the target answer is *airport*. The positive path is: `bar` –[is at location]–> `airport` . The negative path is: `travel` <–[is used for]– `x` –[is at location]– `airport` . We can roughly see that the positive KGs/paths are useful for predicting the correct answer, and vice versa. However, as shown in [39], the model's judgment of KG/path usefulness may not always align with human judgment. The appendix has more examples of coarse/fine explanations.

7 Related Work

Creating Model Explanations Many methods aim to explain PLMs' predictions by highlighting important tokens in the model's text input. Such methods are usually gradient-based [43, 23, 9], attention-based [34, 45, 13, 21], or occlusion-based [11, 36, 19, 24]. Similarly, for graph encoders, a number of works use post-hoc optimization to identify important nodes [17, 52] or subgraphs [52] in the graph input. Meanwhile, KG-augmented models' attention weights can be used to explain which parts of the KG are important [25, 12, 28, 47, 50]. These KG explanations can be interpreted as identifying knowledge in the KG that is complementary to the knowledge encoded in the PLM.

Learning From Model Explanations Besides manual inspection, explanations can be used in various ways, like extra supervision or regularization [37, 16, 35, 1], pruned inputs [18, 3, 22], additional inputs [15, 7], and intermediate variables [49, 54, 38]. The most similar work to ours is [37], which proposed training a student model to mimic a teacher model's predictions by regularizing the student model's attention via text explanations created from the teacher model. However, [37] aims to evaluate explanations, while our goal is to improve performance via explanations. To the best of our knowledge, SALKG is the first to supervise KG-augmented models with KG explanations.

References

[1] Jacob Andreas, Dan Klein, and Sergey Levine. Learning with latent language. *arXiv preprint arXiv:1711.00482*, 2017.

[2] Jasmijn Bastings and Katja Filippova. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? *arXiv preprint arXiv:2010.05607*, 2020.

[3] Joost Bastings, Wilker Aziz, and Ivan Titov. Interpretable neural predictions with differentiable binary variables. *arXiv preprint arXiv:1905.08160*, 2019.

[4] Antoine Bosselut and Yejin Choi. Dynamic knowledge graph construction for zero-shot commonsense question answering. *arXiv preprint arXiv:1911.03876*, 2019.

[5] Michael Chen, Mike D'Arcy, Alisa Liu, Jared Fernandez, and Doug Downey. CODAH: An adversarially-authored question answering dataset for common sense. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*, pages 63–69, Minneapolis, USA, June 2019. Association for Computational Linguistics.

[6] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. Neural natural language inference models enhanced with external knowledge. *arXiv preprint arXiv:1711.04289*, 2017.

[7] John D Co-Reyes, Abhishek Gupta, Suvansh Sanjeev, Nick Altieri, Jacob Andreas, John DeNero, Pieter Abbeel, and Sergey Levine. Guiding policies with language via meta-learning. *arXiv preprint arXiv:1811.07882*, 2018.

[8] Ernest Davis and Gary Marcus. Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, 58(9):92–103, 2015.

[9] Misha Denil, Alban Demiraj, and Nando De Freitas. Extraction of salient sentences from labelled documents. *arXiv preprint arXiv:1412.6815*, 2014.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[11] Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C Wallace. Eraser: A benchmark to evaluate rationalized nlp models. *arXiv preprint arXiv:1911.03429*, 2019.

[12] Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. Scalable multi-hop relational reasoning for knowledge-aware question answering. *arXiv preprint arXiv:2005.00646*, 2020.

[13] Reza Ghaeini, Xiaoli Z Fern, and Prasad Tadepalli. Interpreting recurrent and attention-based neural models: a case study on natural language inference. *arXiv preprint arXiv:1808.03894*, 2018.

[14] David Gunning. Machine common sense concept paper. *arXiv preprint arXiv:1810.07528*, 2018.

[15] Peter Hase and Mohit Bansal. When can models learn from explanations? a formal framework for understanding the roles of explanation data. *arXiv preprint arXiv:2102.02201*, 2021.

[16] Peter Hase, Shiyue Zhang, Harry Xie, and Mohit Bansal. Leakage-adjusted simulatability: Can models generate non-trivial explanations of their behavior in natural language? *arXiv preprint arXiv:2010.04119*, 2020.

[17] Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, Dawei Yin, and Yi Chang. Graphlime: Local interpretable model explanations for graph neural networks. *arXiv preprint arXiv:2001.06216*, 2020.

[18] Sarthak Jain, Sarah Wiegreffe, Yuval Pinter, and Byron C Wallace. Learning to faithfully rationalize by construction. *arXiv preprint arXiv:2005.00115*, 2020.

[19] Akos Kádár, Grzegorz Chrupała, and Afra Alishahi. Representation of linguistic form and function in recurrent neural networks. *Computational Linguistics*, 43(4):761–780, 2017.

[20] Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. QASC: A dataset for question answering via sentence composition. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8082–8090. AAAI Press, 2020.

[21] Jaesong Lee, Joong-Hwi Shin, and Jun-Seok Kim. Interactive visualization and manipulation of attention-based neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 121–126, 2017.

[22] Tao Lei, Regina Barzilay, and Tommi Jaakkola. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*, 2016.

[23] Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. Visualizing and understanding neural models in nlp. *arXiv preprint arXiv:1506.01066*, 2015.

[24] Jiwei Li, Will Monroe, and Dan Jurafsky. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*, 2016.

[25] Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. KagNet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of EMNLP-IJCNLP*, pages 2829–2839, Hong Kong, China, November 2019. Association for Computational Linguistics.

[26] Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online, November 2020. Association for Computational Linguistics.

[27] Ye Liu, Yao Wan, Lifang He, Hao Peng, and Philip S Yu. Kg-bart: Knowledge graph-augmented bart for generative commonsense reasoning. *arXiv preprint arXiv:2009.12677*, 2020.

[28] Ye Liu, Tao Yang, Zeyu You, Wei Fan, and Philip S Yu. Commonsense evidence generation and injection in reading comprehension. *arXiv preprint arXiv:2005.05240*, 2020.

[29] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[30] Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8449–8456, 2020.

[31] Kaixin Ma, Jonathan Francis, Quanyang Lu, Eric Nyberg, and Alessandro Oltramari. Towards generalizable neuro-symbolic systems for commonsense question answering. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 22–32, Hong Kong, China, November 2019. Association for Computational Linguistics.

[32] Gary Marcus. Deep learning: A critical appraisal. *arXiv preprint arXiv:1801.00631*, 2018.

[33] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[34] Akash Kumar Mohankumar, Preksha Nema, Sharan Narasimhan, Mitesh M Khapra, Balaji Vasan Srinivasan, and Balaraman Ravindran. Towards transparent and explainable attention models. *arXiv preprint arXiv:2004.14243*, 2020.

[35] Sharan Narang, Colin Raffel, Katherine Lee, Adam Roberts, Noah Fiedel, and Karishma Malkan. Wt5?! training text-to-text models to explain their predictions. *arXiv preprint arXiv:2004.14546*, 2020.

[36] Nina Poerner, Benjamin Roth, and Hinrich Schütze. Evaluating neural network explanation methods using hybrid documents and morphological agreement. *arXiv preprint arXiv:1801.06422*, 2018.

[37] Danish Pruthi, Bhuwan Dhingra, Livio Baldini Soares, Michael Collins, Zachary C Lipton, Graham Neubig, and William W Cohen. Evaluating explanations: How much do explanations from the teacher aid students? *arXiv preprint arXiv:2012.00893*, 2020.

[38] Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. Explain yourself! leveraging language models for commonsense reasoning. *arXiv preprint arXiv:1906.02361*, 2019.

[39] Mrigank Raman, Aaron Chan, Siddhant Agarwal, Peifeng Wang, Hansen Wang, Sungchul Kim, Ryan Rossi, Handong Zhao, Nedim Lipka, and Xiang Ren. Learning to deceive knowledge graph augmented models via targeted perturbation. *arXiv preprint arXiv:2010.12872*, 2020.

[40] Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pages 4967–4976, 2017.

[41] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer, 2018.

[42] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: an open multilingual graph of general knowledge. In *Proceedings of AAAI*, pages 4444–4451, 2017.

[43] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR, 2017.

[44] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. 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 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[45] Martin Tutek and Jan Šnajder. Staying true to your word:(how) can attention become explanation? *arXiv preprint arXiv:2005.09379*, 2020.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[47] Peifeng Wang, Nanyun Peng, Pedro Szekely, and Xiang Ren. Connecting the dots: A knowledgeable path generator for commonsense question answering. *arXiv preprint arXiv:2005.00691*, 2020.

[48] Xiaoyan Wang, Pavan Kapanipathi, Ryan Musa, Mo Yu, Kartik Talamadupula, Ibrahim Abdelaziz, Maria Chang, Achille Fokoue, Bassem Makni, Nicholas Mattei, et al. Improving natural language inference using external knowledge in the science questions domain. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7208–7215, 2019.

[49] Sarah Wiegreffe, Ana Marasovic, and Noah A Smith. Measuring association between labels and free-text rationales. *arXiv preprint arXiv:2010.12762*, 2020.

[50] Jun Yan, Mrigank Raman, Aaron Chan, Tianyu Zhang, Ryan Rossi, Handong Zhao, Sungchul Kim, Nedim Lipka, and Xiang Ren. Learning contextualized knowledge structures for commonsense reasoning. *arXiv preprint arXiv:2010.12873*, 2020.

[51] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. *arXiv preprint arXiv:2104.06378*, 2021.

[52] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. In *Advances in neural information processing systems*, pages 9244–9255, 2019.

[53] Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. Commonsense knowledge aware conversation generation with graph attention. In *IJCAI*, pages 4623–4629, 2018.

[54] Wangchunshu Zhou, Jinyi Hu, Hanlin Zhang, Xiaodan Liang, Maosong Sun, Chenyan Xiong, and Jian Tang. Towards interpretable natural language understanding with explanations as latent variables. *arXiv preprint arXiv:2011.05268*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]