Predicting ConceptNet Path Quality Using Crowdsourced Assessments of Naturalness

Yilun Zhou
MIT CSAIL
yilun@mit.edu

Steven Schockaert
Cardiff University
SchockaertS1@cardiff.ac.uk

Julie A. Shah
MIT CSAIL
julie_a_shah@csail.mit.edu

ABSTRACT

In many applications, it is important to characterize the way in which two concepts are semantically related. Knowledge graphs such as ConceptNet provide a rich source of information for such characterizations by encoding relations between concepts as edges in a graph. When two concepts are not directly connected by an edge, their relationship can still be described in terms of the paths that connect them. Unfortunately, many of these paths are uninformative and noisy, which means that the success of applications that use such path features crucially relies on their ability to select high-quality paths. In existing applications, this path selection process is based on relatively simple heuristics. In this paper we instead propose to learn to predict path quality from crowdsourced human assessments. Since we are interested in a generic task-independent notion of quality, we simply ask human participants to rank paths according to their subjective assessment of the paths' *naturalness*, without attempting to define naturalness or steering the participants towards particular indicators of quality. We show that a neural network model trained on these assessments is able to predict human judgments on unseen paths with near optimal performance. Most notably, we find that the resulting path selection method is substantially better than the current heuristic approaches at identifying meaningful paths.

CCS CONCEPTS

• **Information systems** → **Crowdsourcing**; *Answer ranking*; • **Computing methodologies** → **Knowledge representation and reasoning**; *Natural language processing*.

KEYWORDS

Crowdsourcing, Knowledge Graph, Feature Selection, Commonsense Knowledge, ConceptNet

ACM Reference Format:
Yilun Zhou, Steven Schockaert, and Julie A. Shah. 2019. Predicting ConceptNet Path Quality Using Crowdsourced Assessments of Naturalness. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313486

1 INTRODUCTION

Many applications require information about the semantic relation between two (or more) words, concepts, or entities. For example, a recommender system needs to recommend items related to a user's browsing history, a task allocation agent needs to match people's experiences and skills to a pool of problems to maximize problem-solving efficiency, and a household robot given the instruction to "wash the plates" needs to infer that a dishwasher could be used. Open-domain knowledge graphs such as ConceptNet (27) allow us to characterize such semantic relationships in the form of relational paths. Compared to the use of word embeddings (21) for characterizing relatedness, knowledge graphs have the potential advantage of producing easier-to-understand characterizations, and they can capture relationships that go beyond what is encoded in standard word embeddings (33).

Typical knowledge graphs (KGs), such as DBpedia and WikiData, are concerned with named entities and their relations (e.g. "Abraham Lincoln" is one of "United States Presidents"). In this paper, however, we are concerned with capturing semantic relationships between common nouns or concepts, which requires a commonsense KG such as ConceptNet. Despite its indisputable value, however, effectively using ConceptNet in applications comes with a unique set of challenges. The knowledge captured in ConceptNet is, by design, often informal, subjective and vague. Due to the fact that ConceptNet partly consists of unverified crowdsourced assertions, it is also noisier than many other KGs. Furthermore, many commonsense assertions are true only under some circumstances and to some extent. For example, ConceptNet encodes that "popcorn" is required for "watching a movie", which captures the useful commonsense knowledge that eating popcorn is associated with watching a movie, but the statement that popcorn is *required* is nonetheless false. In addition, ConceptNet only disambiguates concepts to a coarse part-of-speech level (e.g. noun meaning vs verb meaning of the word "watch"). Finally, a lot of concepts are linked by the generic "RelatedTo" relation, which covers relationships as diverse as collocations ("Tire $\overset{\text{RelatedTo}}{\longleftrightarrow}$ Spare"), hypernyms ("Tire $\overset{\text{RelatedTo}}{\longleftrightarrow}$ All Seasons Tire"), co-meronyms ("Tire $\overset{\text{RelatedTo}}{\longleftrightarrow}$ Exhaust"), homonyms ("Tire $\overset{\text{RelatedTo}}{\longleftrightarrow}$ Tier"), and very loosely related terms ("Tire $\overset{\text{RelatedTo}}{\longleftrightarrow}$ Clothes").

Because of those challenges, few authors use relational paths from ConceptNet in applications. In particular, while several authors have found the knowledge encoded in ConceptNet to be highly valuable, they typically restrict themselves to using relationships that are directly encoded as an edge. For example, Speer et al. (27) showed that ConceptNet can be used to improve word embeddings by incorporating the intuition that words which are linked by an edge in ConceptNet should be represented by similar vectors. We believe that this common restriction to direct edges is due to the lack of sufficiently accurate methods for filtering nonsensical paths, or conversely, for identifying the most natural paths. This path selection problem is the focus of our paper.

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313486

In existing work (11, 19), the problem of selecting high-quality ConceptNet paths has been addressed in a heuristic way (see Section 4). While some intuitions about high-quality paths can easily be formulated (e.g. shorter paths tend to be more informative than longer paths, the nodes occurring in natural paths tend to have similar word vector representations), such heuristic methods still fail to filter out many nonsensical paths, and conversely sometimes erroneously filter out highly valuable paths. For example, the best path between "lead" and "poison" found by one of the standard heuristics (pairwise baseline, see Section 4.3) is "Lead $\xleftrightarrow{\text{Synonym}}$ Take $\xleftrightarrow{\text{DistinctFrom}}$ Give $\xleftrightarrow{\text{RelatedTo}}$ Poison", which uses the verb meaning of "lead", despite the fact that the noun meaning (i.e. the poisonous chemical element Pb) is more relevant in this case.

Rather than trying to construct increasingly intricate heuristics, we propose to learn to predict path quality using a neural network model. To this end, we rely on crowdsourced assessments about the *naturalness* of ConceptNet paths. Specifically, to collect training data, human annotators were asked to choose the more natural path among a pair of paths, without any guidance on how to interpret naturalness. This notion of naturalness was chosen because it is intuitively easy to understand for crowdsourcers (as opposed to e.g. terms such as "semantic coherence" or "predictive value"). The term is also deliberately vague, as we do not want to steer annotators towards particular types of features. The resulting pairwise judgments are then used to train a neural network to predict a latent naturalness score, which can be used for path ranking or path selection.

The main contribution of our work is to show that people's intuitive understanding of naturalness is sufficiently coherent to be used as training data for a data-driven path selection method. To this end, we train a simple neural network model to predict latent naturalness scores that are predictive of human pairwise judgments. We find that our model can predict such judgments with a performance that is close to the optimum suggested by the inter-annotator agreement (Sec 4.2). For example, the best path found by our model for the above example is "Lead $\xrightarrow{\text{HasProperty}}$ Toxic $\xleftrightarrow{\text{RelatedTo}}$ Lethal $\xleftrightarrow{\text{RelatedTo}}$ Poison". We also show, for a number of different evaluation tasks, that our method allows us to select semantically more meaningful paths than the previously proposed heuristics.

2 RELATED WORK

Knowledge graphs: KGs explicitly encode relationships between different entities as subject-predicate-object triples. Such triples can be seen as defining a graph, where the subject and object entities refer to nodes of the graph and the predicates correspond to edge labels. KGs are typically constructed by a domain expert, via crowdsourcing, or by extracting assertions from a natural language corpus (6). In this work, we used ConceptNet (20), one of the most comprehensive commonsense knowledge graphs with 46 relation types, nearly 1 million concepts (words and phrases), and nearly 3 million edges (i.e. triples). ConceptNet partly obtained through crowdsourcing, but also incorporates external sources such as OpenCyc (18), WordNet (22), Verbosity (29), DBPedia (2), and Wiktionary[1].

Knowledge base construction by crowdsourcing: While the use of crowdsourcing for constructing knowledge bases has already been well studied, most existing approaches focus on knowledge acquisition. This can involve, for instance, direct collaborative editing of a knowledge base (3, 31) or indirect construction of a knowledge base through the use of an interactive game (29). In contrast to these works, our focus is on using crowdsourcing for learning how to detect noisy paths within an existing knowledge graph.

An important challenge with crowdsourcing is the fact that there will inevitably be disagreements within the collected data. One framework (1) suggests that this can be due to (i) the inclusion of low-quality participants, (ii) an ambiguous interpretation of the input data and (iii) an ambiguous definition of the labels that crowdsourcers are required to provide. In our work, we mitigated the first issue with a quality control mechanism (Section 4.1). To address the remaining two issues, we will rely on a probabilistic generative model to interpret the provided ratings.

Word embeddings: Word embedding methods (21, 24) represent each word in a relatively low-dimensional space of typically around 300 dimensions, which is estimated from word co-occurrence statistics. With most training procedures, vector differences represent abstract relations in the resulting embedding space: for example, $\mathbf{v}_{\text{king}} - \mathbf{v}_{\text{queen}} \approx \mathbf{v}_{\text{man}} - \mathbf{v}_{\text{woman}}$, in which $\mathbf{v}_{\text{word}}$ represents the embedding of a given word.

Both KGs and word embeddings capture aspects of meaning, and can thus to some extent be seen as alternatives. One advantage of word embeddings is that they capture types of knowledge which are difficult to encode symbolically. For example, the cosine similarity between word vectors tends to correlate very strongly with human perceptions of word similarity. On the other hand, although related words are often close to each other in the embedding space, it is difficult to "reverse-engineer" and describe the specific relation from the vector difference (5). Moreover, when multiple relations exist between two concepts (e.g. "France $\xleftrightarrow{\text{BorderWith}}$ Germany" and "France $\xleftrightarrow{\text{AllyWith}}$ Germany"), the vector difference can only reflect an aggregate. Apart from issues which arise because of the use of vectors, there are also some problems that are more generally related to the use of co-occurrence statistics for representing meaning. As a simple illustration of such problems, a system (16) incorrectly assumes that garlic has wings, because of the prevalence of phrases such as "garlic chicken wings".

Given the complementary nature of the KGs and word embeddings, it is perhaps not surprising that several studies have found it to be beneficial to integrate these two types of resources. Some studies (e.g., (10, 27, 33)) reported that knowledge graphs can be used to improve embeddings to achieve better results on various benchmark tasks such as synonym selection and analogy question solving. Conversely, embeddings can also be incorporated into methods that rely on paths in knowledge graphs, for example to cluster such paths (11), or as features for a path selection method, as in our work.

Path Features in Applications: Several prior works have described systems that incorporated knowledge graph paths as features. For example, Boteanu and Chernova (4) used ConceptNet to solve analogy questions (e.g., "dog is to animal as banana is to fruit") by comparing similarities between paths. Lin et al. (19) proposed models for learning the embeddings of paths and showed that they performed better on problems such as relation prediction compared to embeddings computed by previous methods. Guu et al. (12) proposed to answer

[1] https://www.wiktionary.org/

compositional queries (e.g., "Q: What is the population of the capital of Russia?" "A: Russia $\xrightarrow{\text{Capital}}$ Moscow $\xrightarrow{\text{Population}}$ 12.2 million") by traversing the knowledge graph in *vector space*, representing traversal as a series of matrix-vector multiplications. Das et al. (8) proposed a recurrent neural network (RNN) model with attention mechanism to reason over entities and relations in a knowledge graph, and observed improved performance in path query answering upon that by Guu et al. (12).

Among many applications, knowledge graph completion (i.e., inferring missing relations from existing paths) has been a popular target. Lao and Cohen (17) used path features for knowledge graph completion in the path ranking algorithm (PRA). Gardner et al. (11) incorporated embeddings of the *relations*, allowing the system to recognize semantically similar relations, such as "(river) runs through (city)" and "(river) flows through (city)." Neelakantan et al. (23) built upon the same PRA idea, but used an RNN to model paths. Toutanova et al. (28) proposed a dynamic programming algorithm that can incorporate both edge and vertex features and reported better performance on knowledge graph completion. While all of the above-mentioned knowledge graph completion works use some version of PRA to generate paths, Xiong et al. (32) trained a reinforcement learning agent for path generation that rewards accuracy, efficiency and diversity, and demonstrated better performance than PRA-generated paths.

Path Selection: The number of paths between two nodes in a knowledge graph tends to grow exponentially with path length. In addition, while each type of path can be viewed as encoding a kind of semantic relationship, for many paths this relationship is difficult to interpret. Thus, all of the works described above mitigate the scalability and/or quality problem via heuristics. To limit the number of paths, Boteanu and Chernova (4) stopped the path search after a pre-defined number of nodes are explored. Lao and Cohen (17) and Guu et al. (12) limited the maximum path length. To obtain natural paths, Gardner et al. (11) favored paths that contain words with higher embedding similarity. Lin et al. (19) calculated the quality of paths using a heuristic based on vertex degree and network flow.

We note that while ConceptNet has been proven useful, most works only use direct edges, with one work (4) being a notable exception. For other types of knowledge graphs, it was already shown that incorporating longer relational paths can be highly beneficial, and there is no reason to assume that this situation would be any different for ConceptNet. For instance, while there is an intuitively obvious semantic relationship between the words "beach" and "sun", there is no direct edge connecting these words in ConceptNet. However, this relationship can be uncovered by observing that ConceptNet contains the path "Beach $\xleftarrow{\text{RelatedTo}}$ Sunbathing $\xleftarrow{\text{RelatedTo}}$ Sun". Nevertheless, to enable more effective use of such relational paths from ConceptNet, we believe that more work is needed on how to avoid nonsensical paths, which is the focus of this paper.

3 METHOD

3.1 Problem Formulation

Our goal in this work is to train a classifier for predicting path quality based on crowdsourced information. Specifically, we ask human annotators to assess the *naturalness* of ConceptNet paths. Rather than trying to provide guidelines on how naturalness should be understood, we simply rely on annotators' intuitive understanding of this notion. This has several advantages, including the fact that we do not steer annotators towards particular indicators and the fact that this makes the annotation task much easier. Moreover, given that we are interested in a task-independent form of path quality (i.e., our focus is on eliminating non-sensical and uninformative paths, rather than on selecting the best paths for a particular application), it is not clear what further guidance would be meaningful for annotators. As we will discuss in more detail in Section 4.2, despite the lack of guidance on how naturalness should be understood, we observed a high inter-annotator agreement in practice.

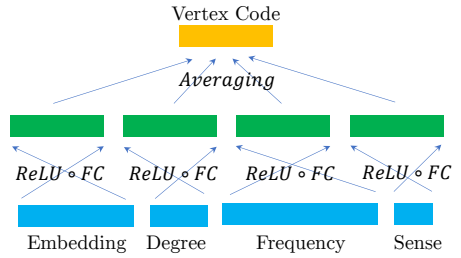
Clearly, however, asking participants to provide ratings on an absolute scale is problematic: people have varying thresholds for naturalness, and these thresholds may also change over time. After observing a large number of unnatural paths, people may lower their thresholds to be more willing to accept a path as natural, and vice versa. Therefore, we instead asked participants about pairwise comparisons: determining which of two given paths is more natural. In this setting, if one path is more natural than the other, two raters would provide the same answer even if they maintained different absolute naturalness thresholds, or if they shifted their underlying absolute threshold unconsciously over time.

If the two paths are equally (un)natural, the selected answer would be more or less arbitrary, and even the same annotator might not consistently give the same answer when presented with the same pair more than once. Therefore, in our model, we assume that answers to these pairwise comparison questions are probabilistically generated from a latent naturalness score. Specifically, each path is assumed to have a latent naturalness score m, such that for a pair paths with scores m_1 and m_2, the observed answer is drawn from a Bernoulli distribution, with probability $e^{m_1}/(e^{m_1} + e^{m_2})$ that the first path is selected. The problem we consider is to learn a model that can predict the naturalness score m of a given path, using crowdsourced answers of pairwise comparisons as supervision signal.

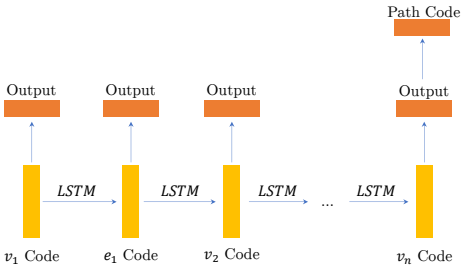
3.2 Model

In the big picture, our model consists of an encoder and a pairwise predictor. The encoder is a long short-term memory (LSTM) network (13) that transforms a path into a code vector. The predictor then takes the path codes and computes the probability that one path is more natural than the other. Each path is represented by an alternating sequence of vertex and edge representations, $(v_1, e_1, v_2, e_2, ..., e_{n-1}, v_n)$. Each vertex representation is a list of n_v features, $v_i = (\mathbf{v}_{i,1}, \mathbf{v}_{i,2}, ..., \mathbf{v}_{i,n_v})$, with j-th feature being a d_j^v-dimensional vector. The edge representations are structured similarly. The features which we use are summarized in Section 3.4.

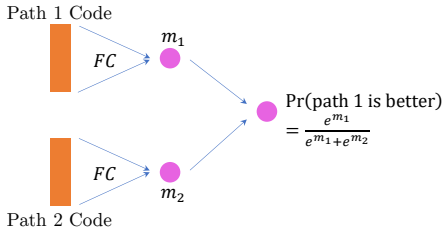
A standard LSTM architecture can only process sequential data of the same dimension. Thus, vertex and edge representations must first pass through an encoder. Figure 1a depicts how a neural network encodes a vertex into a vertex code of pre-defined length l_f. Each feature (such as the word embedding and absolute frequency in some large corpus) is transformed through a rectified linear unit (ReLU)-activated fully-connected (FC) layer to a vector of length l_f. The overall code is the average of the vectors. The encoding of an edge to the same length l_f is conducted in a similar fashion. Figure



(a) The encoding of a vertex.



(b) The encoding of the path.



(c) The predictor model.

Figure 1: LSTM model architecture. Blue (bottom) cells represent raw features (of variable lengths). Green (middle) cells represent transformed features (of length l_f). Orange (top) cells represent the final code for the vertex (of length l_f). The encoding of an edge is computed in a similar fashion. The codes (yellow) for vertices and edges are successively processed by an LSTM network. The last state $\mathbf{h}_{2n-1}$ (orange) is used as the code for the entire path. The path codes (orange) first pass separately through fully-connected layers (with shared weight) with an output dimension of 1. A softmax layer then calculates the probability that path 1 is more natural than path 2.

1b depicts the way in which the path code is calculated. After the codes for each vertex and edge are computed, they are sequentially aggregated by an LSTM network. The last state $\mathbf{h}_{2n-1}$ is interpreted as the code of the path. As shown in Figure 1c, the predictor then outputs the pairwise comparison result. It first transforms each path code to a score (m_1 and m_2) using linear layers with shared weight. The softmax function is then applied to the two scores to calculate a probability. While the network predicts pairwise comparisons, it is the score m_i produced by the network – or, at least, the ordering induced by the scores – that matters for applications.

The problem we consider falls under the broad umbrella of learning to rank, for which many algorithms have already been proposed. We use an LSTM encoder because it elegantly deals with variable length input, while most other methods can only take fixed length input representations.

3.3 Training

We used the negative-log-likelihood (NLL) loss as our objective function, along with the Adam optimization algorithm (14). Although the neural network model seems like a natural choice for the data generation process described in Section 3.1, here we show that the predicted scores m indeed converge to the latent scores from which the pairwise comparison results were generated, up to a constant difference.

Consider two paths with scores m_1 and m_2. According to our generative model, the first path is selected with probability $p_{m_1 m_2} = e^{m_1}/(e^{m_1}+e^{m_2})$. Note that from $e^{m_1}/(e^{m_1}+e^{m_2}) = e^{m'_1}/(e^{m'_1}+e^{m'_2})$ it follows that $m_1 - m'_1 = m_2 - m'_2$ by simple algebra. This means that if our system can correctly predict the probabilities $p_{m_1 m_2}$, for any pair of paths, then it must be the case that there is a fixed constant c such that for each path with naturalness score m, the predicted naturalness score m' is such that $m = m' + c$.

It remains to be shown that the NLL objective will indeed lead our neural network model to predict the correct probabilities. Specifically, assume that observations $\{y_i \sim \text{Bernoulli}(q_\theta(\mathbf{x}_i))\}_{i=1}^n$ are given, where $\mathbf{x}_i$ is the feature representation of a given pair of paths and $q_\theta(\mathbf{x}_i)$ is the corresponding output of our neural network model when the parameters are set as θ. We show that minimizing the NLL objective will result in parameters ϕ such that $q_\theta = q_\phi$. For any fixed $\mathbf{x}_i$, given a set of observations $\{(\mathbf{x}_i, y_{i,1}), ..., (\mathbf{x}_i, y_{i,n_i})\}$, we define the expected data likelihood with respect to q_i to be as follows:

$$\mathbb{E}_{q_i}\left[\Pr(y_{i,1:n_i}|\mathbf{x}_i)\right] = \mathbb{E}\left[\prod_{j=1}^{n_i}\left[y_{i,j}q_i + (1-y_{i,j})(1-q_i)\right]\right].$$

When $y_{i,1:n_i}$ is generated by Bernoulli $(q_\theta(\mathbf{x}_i))$, it can be shown that $\arg\max_{q_i} \mathbb{E}_{q_i}\left[\Pr(y_{i,1:n_i}|\mathbf{x})\right] = q_\theta(\mathbf{x}_i)$. For the entire dataset $\{(\mathbf{x}_1, y_1), .., (\mathbf{x}_n, y_n)\}$, since the y_i's are independent of each other given $\mathbf{x}_i$, the expected joint likelihood for the entire dataset

$$\mathbb{E}_{q_\phi}\left[\Pr(y_{1:n}|\mathbf{x}_{1:n})\right] = \mathbb{E}\left[\prod_{j=1}^{n}\left[y_j q_\phi(\mathbf{x}_j) + (1-y_j)(1-q_\phi(\mathbf{x}_j))\right]\right]$$

is then maximized for $q_\phi(\mathbf{x}) = q_\theta(\mathbf{x})$ for all $\mathbf{x}$. Therefore, we can simply minimize NLL (i.e. $-\log\left[\Pr(y_{1:n}|\mathbf{x}_{1:n})\right]$).

Note in particular that the network will converge to optimal predictions even with only one answer for each pair, which we can exploit to maximize the diversity of the crowdsourced annotations: by only collecting a single annotation for each pair of paths, a broader range of paths can be assessed (with a fixed budget) than with multiple annotations per pair. While each individual answer will to some extent be arbitrary (e.g., the fact that path 1 was selected could mean that path 1 is more natural or that both paths are approximately equally natural and that path 1 was selected by chance), because the network

is trained on many different pairs, it will still learn to differentiate between clear-cut cases and borderline cases.

3.4 Features

In this section, we describe the features we used to encode vertices and edges. The number in parenthesis after each feature name indicates the dimension of that feature.

Vertex embedding (300) This feature is taken directly from the 300-dimensional GloVe (24) embedding, pre-trained on the Common Crawl[2] dataset with 840 billion tokens. In some experiments, we used principal component analysis (PCA) to first reduce the dimensionality of this feature before inputting it into the neural network.
Vertex frequency (1) This feature is a scalar representing the frequency of the given word. We estimated the (unnormalized) frequency using Zipf's law (35) from word occurrence ranks, which can be derived from the pre-trained GloVe embedding since it orders words by frequency. For example, the word "science" is ranked 717th, and is thus assigned a frequency of 1/717=1.39e-3.
Vertex degree (1) This feature is the number of neighbors (in both directions) of the vertex in the graph, representing how well connected the word is within ConceptNet.
Vertex sense score (1) This feature is a number between 0 and 1 representing how consistent is the overall meaning of the vertex compared to its neighbors on the left and right. For example, the sense score for the word "book" in the path "Knowledge $\xleftarrow{\text{HasA}}$ Book $\xleftrightarrow{\text{RelatedTo}}$ Restaurant" would be quite low because it is used in two different senses . On the other hand, the sense score for "book" in "Knowledge $\xleftarrow{\text{HasA}}$ Book $\xleftrightarrow{\text{RelatedTo}}$ Paper" would be high. We believe that a lower sense score would result in the path being considered less natural. In calculating this feature, we borrow and extend the idea of "word sense profile" proposed by (7). The details of the sense score calculation can be found in the appendix of the extended version[3].
Edge ends similarity (1) For an edge connecting two vertices (s, t), this feature is the cosine similarity of the embeddings of s and t.
Edge direction (3) This feature is a one-hot vector that represents if the edge is forward ("Dog $\xrightarrow{\text{IsA}}$ Animal"), backward ("Wheel $\xleftarrow{\text{HasA}}$ Car"), or bidirectional ("Big $\xleftrightarrow{\text{Antonym}}$ Small").
Edge relation (46) This feature is a one-hot vector that represents the relation type of the edge.
Edge provenance (6) In ConceptNet, each edge is derived from at least one source. In the case of multiple sources, the weight is the sum of the weights across all sources. However, weights across sources are not directly comparable; for example, nearly all edges from WordNet have a weight of 2.0, while those from Wiktionary have a weight of 1.0. This does not necessarily mean that WordNet is more reliable than Wiktionary. Thus, we encode the provenance of an edge using a 6-dimensional vector, with one component for each possible source: WordNet, DBpedia, Verbosity, Wiktionary, OpenCyc, and Open Mind Common Sense (26). When an edge has only one source, its component in the vector has a value equal to the weight, and all other entries are 0. When an edge has multiple associated sources, we split the weight across the sources using the most common weight of each source (e.g., 2.0 for WordNet).
Edge sense score (1) Similar to the vertex sense score, this feature is also a number between 0 and 1 characterizing the consistency of meaning, but at an edge level. Again, the details are included in the extended version[3].

4 EXPERIMENTS

In this section, we first describe our data collection procedure. Then we present an evaluation of the collected data and learned model.[4]

4.1 Data Collection

For our experiments, we used Amazon Mechanical Turk to collect pairwise human rating data. Each questionnaire consisted of 73 pairs, including 60 genuine pairs and 13 quality-control pairs. The quality-control pairs consisted of an obviously good path and an obviously bad path. The good paths were manually specified and meant to be very natural and straightforward (e.g., Email $\xrightarrow{\text{UsedFor}}$ Communication $\xleftarrow{\text{UsedFor}}$ Telephone). To generate bad paths, words and relations were sampled independently at random (e.g., "Beautiful $\xrightarrow{\text{IsA}}$ Other $\xleftrightarrow{\text{RelatedTo}}$ Them $\xleftarrow{\text{MotivatedBy}}$ Rug"). We only used answers from questionnaires for which all 13 quality-control questions were answered correctly. The genuine pairs consisted of randomly sampled paths from ConceptNet. We used a number of different strategies to sample these paths, as outlined below. Overall, around 1,500 valid responses were collected, giving us around 90,000 pairs of paths to use in different settings of the experiment.

4.2 Inter-Annotator Agreement

The human annotators were provided with a generic question: "which of the two paths is more natural?" Different participants may interpret this question slightly differently, focusing on different aspects of naturalness. Thus, one important question is how much humans agree with each other about naturalness. Establishing the inter-annotator agreement level is also useful because it serves as an intrinsic, model-agnostic performance upper bound.

Our default data collection strategy is to obtain only one answer for each pair of paths. This strategy maximizes the coverage of our dataset, while still allowing the model to make probabilistic predictions, as discussed in Section 3.3. However, to evaluate the inter-annotator agreement, we also collected a multi-response dataset, with 59 questions each answered by 16 different participants.

Table 1 summarizes the results of an analysis of this dataset. The first column lists the different possible opinion splits for a given question; for example, 13/3 corresponds to a question for which 13 people preferred one of the paths, and 3 people preferred the other. The second column shows how many of the 59 questions have a given opinion split.

The last two columns of Table 1 refer to results obtained by our model trained in the 1st setting described below in Section 4.3. The third column shows the number of questions predicted correctly, i.e. the number of questions for which the prediction of our model coincides with the majority opinion of the human participants.

[2] http://commoncrawl.org/
[3] https://arxiv.org/pdf/1902.07831.pdf
[4] The code and data for the experiments can be found at https://github.com/YilunZhou/path-naturalness-prediction

The last column is the model's average confidence in the majority consensus. The 8/8 row does not include prediction statistics because there is no majority consensus for these questions.

A number of conclusions can be drawn from these results. First, given the disagreement levels shown in the second column, the theoretically best performing model, which always predicts the majority preference, would achieve 70.1% accuracy on this dataset. This shows that while people do not agree with each other on all of the questions, for the majority of pairs, human annotators do have a clear preference. Note that we should not expect a perfect agreement, since in some cases both of the paths may be equally natural or equally unnatural. This is illustrated in Table 2, which shows examples of pairs with different opinion splits. As can be seen from these examples, the fact that there is no majority for a given pair does not necessarily mean that the paths involved are of low quality. It simply means that the two paths are approximately equally natural.

As can be seen from the third column in Table 1, when there is a clear human consensus about which of the two paths is most natural, our model can predict this with a very high accuracy; e.g. for all questions where the opinion split was 14/2 or better, the majority view was predicted correctly. Furthermore, there is a strong positive correlation between the confidence scores predicted by our model, shown in the last column, and the amount of disagreement among human annotators. Our model is thus able to distinguish between cases where the difference in naturalness is clear-cut and cases where human annotators would be undecided.

Opinion Split	# Q	# Correct	Avg Conf
8/8	7	NA	NA
9/7	11	5	52.2%
10/6	6	5	60.1%
11/5	8	7	64.4%
12/4	11	7	56.6%
13/3	5	4	70.0%
14/2	5	5	72.5%
15/1	4	4	76.2%
16/0	2	2	89.8%

Table 1: Results for the multi-response dataset.

OS	Paths (The one on top is favored by the majority)
8/8	Forest $\xleftarrow{\text{AtLocation}}$ Country $\xleftrightarrow{\text{RelatedTo}}$ Geography $\xleftrightarrow{\text{RelatedTo}}$ Land
	Kingdom $\xleftrightarrow{\text{Synonym}}$ Land $\xleftarrow{\text{MadeOf}}$ Mountain $\xleftrightarrow{\text{RelatedTo}}$ Hill
	Dream $\xleftrightarrow{\text{RelatedTo}}$ State $\xleftarrow{\text{AtLocation}}$ City $\xleftrightarrow{\text{RelatedTo}}$ Population
	Desk $\xleftrightarrow{\text{RelatedTo}}$ Office $\xleftrightarrow{\text{RelatedTo}}$ Area $\xleftrightarrow{\text{RelatedTo}}$ Science
11/5	Blood $\xrightarrow{\text{AtLocation}}$ Person $\xleftrightarrow{\text{RelatedTo}}$ Home $\xleftrightarrow{\text{Antonym}}$ Street
	Range $\xleftrightarrow{\text{RelatedTo}}$ Food $\xrightarrow{\text{AtLocation}}$ Home $\xleftrightarrow{\text{Antonym}}$ Office
12/4	Kingdom $\xleftrightarrow{\text{RelatedTo}}$ Country $\xleftarrow{\text{AtLocation}}$ City $\xleftarrow{\text{AtLocation}}$ Office
	Sun $\xleftrightarrow{\text{RelatedTo}}$ Source $\xleftrightarrow{\text{RelatedTo}}$ Blood $\xleftrightarrow{\text{RelatedTo}}$ Tissue
13/3	Person $\xleftrightarrow{\text{RelatedTo}}$ Woman $\xleftrightarrow{\text{DistinctFrom}}$ Men $\xleftrightarrow{\text{RelatedTo}}$ People
	Type $\xleftrightarrow{\text{RelatedTo}}$ Unit $\xleftrightarrow{\text{RelatedTo}}$ Card $\xleftrightarrow{\text{RelatedTo}}$ Holiday
16/0	School $\xrightarrow{\text{AtLocation}}$ City $\xleftarrow{\text{IsA}}$ Town $\xrightarrow{\text{AtLocation}}$ Country
	Point $\xleftrightarrow{\text{RelatedTo}}$ Mountain $\xleftrightarrow{\text{RelatedTo}}$ Wave $\xrightarrow{\text{IsA}}$ Woman
	People $\xleftrightarrow{\text{RelatedTo}}$ Life $\xleftarrow{\text{PartOf}}$ Fun $\xleftarrow{\text{IsA}}$ Soccer
	Plane $\xleftrightarrow{\text{Antonym}}$ Point $\xleftrightarrow{\text{RelatedTo}}$ Hand $\xleftrightarrow{\text{RelatedTo}}$ Instrument

Table 2: Pairs of paths with different opinion splits (OS). In each cell the most favored path is shown on top.

4.3 Prediction Accuracy

4.3.1 Experimental Setup. In this set of experiments we studied how well our model can predict human judgements in three settings.

- In the 1st setting, we first determined a set of 100 words. To this end, starting from the center word "science", we performed a random walk on ConceptNet, considering only elementary-school-level nouns[5], until 100 different words had been sampled. We then sampled pairs of paths with 4 or less nodes in the subgraph induced by these 100 words. This dataset included about 10,000 paths. The training set contains 40,000 pairs sampled from 8,000 paths, and the test set contains 1,000 pairs put together from the remaining 2,000 paths. This evaluation allows us to assess the performance of our system in a domain-specific setting, since the model is evaluated on paths involving the same words as those in the training paths (but nonetheless different paths). This performance is recorded in Table 3.
- In the 2nd setting, we collected another set of 100 words using the same method as above, but using "money" as the center word, while additionally ensuring that there is no overlap between this set of 100 words and those from the "science" dataset. We then collected 1,000 paths among those 100 words, and put them into 500 pairs, which we use to evaluate models trained on the "science" dataset. In this way, we can assess the transfer learning performance of our model. This performance is recorded in Table 4.
- In the 3rd setting, we selected the nouns, verbs and adjectives from the 5,000 most frequent English words in the Corpus of Contemporary American English[6], for a total of 3,887 words. We then sampled 80,000 paths of up to 5 nodes, which we split into 20,000 training pairs and 20,000 testing pairs (ensuring that there are no overlapping paths between training and testing sets). Compared with the 1st setting, this represents much wider coverage of the set of words (i.e., embedding space), but much sparser coverage of paths, allowing us to assess the performance of our model in an open-domain setting. This performance is recorded in Table 5.

To put our performance into context, we also considered the following four heuristic baselines, all of which have similar mechanics: computing a score for each path (as does our model), and then selecting the path with the higher score.

Source-Target Baseline (ST-B) scores paths using the cosine similarity between the embeddings of the source and target words (note

[5]http://www.k12reader.com/

[6]https://www.wordfrequency.info/

that the two paths in a pairwise comparison do not typically start and end with the same words);

Pairwise Baseline (Pair-B) scores paths using the average cosine similarity of all word pairs connected with an edge in the path;

Resource Flow Baseline (Flow-B) scores paths using the "path reliability" method proposed by Lin et al. (19) with the idea that a path is better if there are less branches out of the path;

Length Baseline (Path-B) scores paths solely by their length, and favors shorter paths to longer ones.

We performed a grid search of two hyper-parameters: vertex embedding dimension in $\{2, 10, 50, 100, 300\}$ and path code length in $\{1, 2, 5, 10, 20\}$. For the first hyper-parameter, we used principal component analysis (PCA) for dimensionality reduction. We also tested one-hot encodings of words for the 1st setting. For the other settings, this one-hot encoding cannot be used, because it cannot generalize to unseen words (2nd setting) or scale to a large number of words (3rd setting). In addition, Length-B performance is only available for the 3rd setting, because for the first two settings we did not explicitly control for path length. As a result, the vast majority of paths are of the cutoff length of 4, and path length is not predictive.

		Path Code Length				
		1	2	5	10	20
Emb. Dim.	2	62.1	63.1	65.8	65.3	65.6
	10	63.6	65.5	65.6	67.5	66.4
	50	63.7	65.5	66.9	67.4	67.7
	100	65.4	65.9	66.1	**68.2**	67.6
	300	66.2	66.6	67.6	67.6	67.7
One-Hot		67.0	66.4	67.0	67.5	67.9
ST-B				53.4		
Pair-B				58.0		
Flow-B				51.6		

Table 3: Test accuracy (in percentage) in the domain-specific setting of "science" related words.

		Path Code Length				
		1	2	5	10	20
Emb. Dim.	2	60.5	64.6	63.9	65.1	**65.5**
	10	60.1	61.7	64.8	64.9	64.8
	50	62.5	62.8	63.1	63.3	64.0
	100	60.0	61.8	63.1	64.2	64.8
	300	58.8	60.5	61.1	62.0	62.1
ST-B				53.9		
Pair-B				55.9		
Flow-B				52.5		

Table 4: Test accuracy (in percentage) in the transfer learning setting, where the model was trained on the "science" dataset and evaluated on the "money" dataset.

		Path Code Length				
		1	2	5	10	20
Emb. Dim.	2	61.4	60.5	63.1	62.9	63.3
	10	61.2	62.6	65.0	64.9	64.2
	50	61.6	64.5	64.8	65.4	65.8
	100	62.0	64.3	65.0	**67.9**	65.1
	300	61.9	63.0	65.7	65.4	65.4
ST-B				52.3		
Pair-B				57.5		
Flow-B				46.8		
Length-B				55.8		

Table 5: Test accuracy (in percentage) in the open-domain setting of the most frequent English words.

4.3.2 Quantitative Analysis. For the first setting (Table 3), we can see that performance was relatively insensitive to embedding dimension and code length, as long as both were sufficiently high. However, performance on the 2nd setting (Table 4) dropped significantly with higher embedding dimensions, suggesting an overfitting problem, with an embedding dimension of 2 achieving best generalization performance. This suggests that a model that was trained on one domain can indeed successfully be applied to another domain, as long as the embedding dimension is small enough to allow for sufficient generalization. In the 3rd setting (Table 5), we found that a higher embedding dimension is necessary, most likely because the range of meanings is more diverse for this dataset, which includes not only nouns, but also verbs and adjectives.

For all three settings, considering the level of agreement found in Section 4.2, we found that the performance of our model approaches the expected performance upper bound. Moreover, our model performs substantially and consistently better than all of the baselines. In fact, in Table 5 we can see that only one of the baselines is able to outperform the simple path length heuristic (Length-B), and only in a minimal way.

4.3.3 Qualitative Analysis. To qualitatively compare how our model prediction differs from the baselines, Table 6 presents the best paths between the words "health" and "computer" (of up to 4 nodes) selected by our method and the two strongest baselines.

We can see that the top paths found by the pairwise baseline lack variety, most likely due to the high embedding similarity between "health" and "care". In addition, the length baseline does not perform satisfactorily because the shortest paths are not easily understandable without more explanation, with nearly all relations being the generic "$\overset{\text{RelatedTo}}{\longleftrightarrow}$". By contrast, our model (trained on the 3rd open-domain setting) selects paths that overcome both drawbacks. It covers a wider variety of concepts such as "virus" and "expensive" and favors longer paths with smoother transition of node semantics.

In another evaluation, we inspect paths for which our model and the pairwise baseline significantly disagree. We randomly sampled 120,000 paths from the entire ConceptNet graph and use the two methods to predict their quality. Table 7 lists paths that are predicted to be among the top 10% most natural by our model but are in the

Pairwise	Health $\xleftrightarrow{\text{RelatedTo}}$ Care $\xrightarrow{\text{IsA}}$ Work $\xleftrightarrow{\text{RelatedTo}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Care $\xleftrightarrow{\text{RelatedTo}}$ Help $\xleftrightarrow{\text{RelatedTo}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Care $\xleftrightarrow{\text{RelatedTo}}$ Do $\xleftarrow{\text{CapableOf}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Care $\xleftarrow{\text{Desires}}$ Person $\xrightarrow{\text{Desires}}$ Computer
Length	Health $\xleftrightarrow{\text{RelatedTo}}$ System $\xleftrightarrow{\text{RelatedTo}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Level $\xleftrightarrow{\text{RelatedTo}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Apple $\xleftrightarrow{\text{RelatedTo}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Well $\xleftrightarrow{\text{RelatedTo}}$ Screw $\xrightarrow{\text{AtLocation}}$ Computer
Our Model	Health $\xleftarrow{\text{Antonym}}$ Disease $\xleftarrow{\text{Causes}}$ Virus $\xrightarrow{\text{AtLocation}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Care $\xrightarrow{\text{IsA}}$ Work $\xleftrightarrow{\text{RelatedTo}}$ Computer
	Health $\xleftrightarrow{\text{RelatedTo}}$ Insurance $\xrightarrow{\text{HasProperty}}$ Expensive $\xleftarrow{\text{HasProperty}}$ Computer
	Health $\rightarrow$ Sickness $\rightarrow$ Virus $\xrightarrow{\text{AtLocation}}$ Computer

Table 6: Best paths of 4 or less nodes between "health" and "computer" ranked by baselines and our model.

Lead $\xrightarrow{\text{HasProperty}}$ Toxic $\xleftrightarrow{\text{RelatedTo}}$ Toxicant $\xleftrightarrow{\text{Synonym}}$ Poison
Wave $\xrightarrow{\text{IsA}}$ Fluctuation $\xleftrightarrow{\text{RelatedTo}}$ Brainwave $\xrightarrow{\text{DerivedFrom}}$ Brain
Space $\xleftarrow{\text{MadeOf}}$ Passageway $\xrightarrow{\text{AtLocation}}$ Building $\xleftrightarrow{\text{RelatedTo}}$ Station
Food $\xleftrightarrow{\text{RelatedTo}}$ Hechsher $\xleftrightarrow{\text{RelatedTo}}$ Kashrut $\xleftrightarrow{\text{RelatedTo}}$ Law

Table 7: Paths predicted to be among 10% most natural by our model, but 10% least natural by the pairwise baseline.

Bee $\xleftrightarrow{\text{RelatedTo}}$ A $\xleftrightarrow{\text{RelatedTo}}$ After $\xleftrightarrow{\text{RelatedTo}}$ Attack
Space $\xleftrightarrow{\text{RelatedTo}}$ Very $\xleftrightarrow{\text{RelatedTo}}$ Little $\xleftrightarrow{\text{RelatedTo}}$ Moon
Lead $\xleftrightarrow{\text{Synonym}}$ Run $\xleftrightarrow{\text{RelatedTo}}$ Very $\xleftrightarrow{\text{RelatedTo}}$ Poison
Adult $\xleftrightarrow{\text{RelatedTo}}$ A $\xleftrightarrow{\text{RelatedTo}}$ The $\xleftrightarrow{\text{RelatedTo}}$ English

Table 8: Paths predicted to be among 10% most natural by the pairwise baseline, but 10% least natural by our model.

bottom 10% according to the pairwise baseline. Table 8 conversely shows paths that are among the 10% least natural according to our model but the 10% most natural according to the baselines. Paths in both tables were selected at random.

As we can see, the paths in Table 7 are intuitively quite meaningful, with all paths using some rather uncommon but relevant words, which is perhaps clearest in the last example (Hechsher refers to a certification given to a food product that complies with Jewish religious law and Kashrut refers to the set of Jewish dietary laws). By comparison, the paths in Table 8 mostly use very common but uninformative words. This analysis suggests that the pairwise baseline suffers from the so-called hubness problem, i.e. the problem that in high-dimensional vector space embeddings, there are typically a few central objects which are highly similar to many of the other objects (25), an issue which is known to affect word embeddings (9). The pairwise baseline favors paths with very common words, even if they are not semantically related, because these words act as hubs in the word embedding.

4.4 Ablation Study on Feature Importance

To discern the contribution of each feature to the prediction performance, we trained variants of our model with subsets of the full feature set. For this analysis, we used the 3rd evaluation setting with the best performing hyper-parameters (embedding dimension of 100 and path code length of 10). The results are summarized in Table 9.

All Features	67.9%
No vertex embedding	65.4%
No vertex frequency	67.3%
No vertex degree	67.1%
No edge ends similarity	63.5%
No edge direction	67.8%
No edge relation	67.3%
No edge provenance	67.2%
No vertex and edge sense scores	66.9%
Vertex features only	61.6%
Edge features only	63.7%

Table 9: Ablation study on feature importance.

We can see that leaving out the edge ends similarity feature hurts performance most. This is also in accordance with the fact that pairwise baseline in Table 5, which only uses this feature, performs best among the four baseline methods. Furthermore, the second-highest reduction is found when leaving out the vertex embedding feature. Finally, we see that neither vertex features nor edge features alone can achieve optimal prediction performance.

4.5 Naturalness as a Path Selection Criterion

Next we discuss three experiments which are aimed at assessing how well naturalness indicates path quality. We start with an intrinsic evaluation of the semantic coherence of natural paths, after which we discuss two extrinsic tasks: information retrieval and analogy inference.

4.5.1 Semantic Coherence. Let us define the *type* of a path as the sequence of relations appearing in that path, e.g. the type of the path "A $\xrightarrow{\text{IsA}}$ B $\xrightarrow{\text{HasA}}$ C" is ($\xrightarrow{\text{IsA}}$, $\xrightarrow{\text{HasA}}$). In this experiment, we consider pairs of nodes in ConceptNet that are connected with a relational path of a given type, as well as a one-step relation (e.g., "A $\xrightarrow{\text{HasA}}$ C"). In this situation, we call the latter relation a *path-summarizing (PS) relation* for the considered path type. Many path types uniquely determine the PS relation, or at least narrow down the set of candidate PS relations to a small number. For instance, for a path of type ($\xrightarrow{\text{IsA}}$, $\xrightarrow{\text{HasA}}$) we would expect to only see $\xrightarrow{\text{HasA}}$ as the PS relation. However, due to the presence of noise in ConceptNet, some relations of this type will actually have a different PS relation. Motivated by this view, as an intrinsic evaluation task, we propose to assess the extent to which a path selection method is able to select

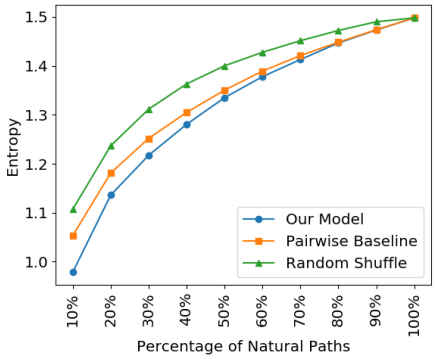


Figure 2: Average entropy of path-summarizing relations for different proportions of natural paths.

semantically coherent paths in terms of the variability of the PS relations among the selected paths.

In particular, for a set of paths, we first determine a mapping from path types to counters of PS relations: $\{p_1 \rightarrow [(r_{11}, c_{11}), (r_{12}, c_{12}), ..., (r_{1n}, c_{1n})], ..., p_m \rightarrow [(r_{m1}, c_{m1}), (r_{m2}, c_{m2}), ..., (r_{mn}, c_{mn})]\}$, in which each p_i is a path type, each r_{ij} is one of 46 relation types from ConceptNet, and each c_{ij} is the number of times the relation r_{ij} was found as a PS relation for path type p_i. The average entropy of the PS relations is then defined as:

$$\frac{-\sum_{i=1}^{m}\left[C_i \cdot \sum_{j=1}^{n} p_{ij} \ln(p_{ij})\right]}{C},$$

where $C_i = \sum_{j=1}^{n} c_{ij}, C = \sum_{i=1}^{m} C_i, p_{ij} = c_{ij}/C_i$. A higher average entropy suggest a higher proportion of spurious PS relations and thus less semantically coherent paths.

Using the 3,887 words from the 3rd setting, we generated 126,600 length-4 paths whose source and target nodes are also directly connected by a relation (i.e. the PS relation). Our model was applied to rank the paths by naturalness. Based on this ranking, we calculate the average entropy of top $N\%$ natural paths, where N varies from 10 to 100, and plot the result in Figure 2. We compared our model to Pair-B, the best-performing baseline for predicting naturalness. The entropy for a random shuffle of the dataset is also depicted for comparison. We see that using our model to select the most natural paths consistently leads to the lowest entropy. Note that all three methods will converge to the same entropy at the 100% mark, at which point all paths are used, regardless of the path selection method.

4.5.2 Information Retrieval. When using a search engine, users often provide under-specified queries, e.g. because they assume that the search engine is "smart" enough to infer their true intention or because they do not actually have a clear idea on how to formulate a better query. Moreover, even if a query is unambiguous, it may use different terms than those appearing in the most relevant documents. To deal with such issues, many information retrieval systems rely on some form of query expansion (34), a strategy for automatically adding semantically related keywords to the user's query. In this experiment, we study the performance of a query expansion method based on natural ConceptNet paths.

We use the TREC 2004 Robust Retrieval Track dataset (30), commonly known as ROBUST04, consisting of 250 queries and around 260,000 documents. Each query from this dataset typically consists of 2 or 3 words, and is associated on average with 1,000 annotated documents (marked as either positive or negative). We use a simple TF-IDF retrieval model, in which the TF-IDF weighting scheme is used to vectorize both the documents and the queries and the cosine similarity is used as the ranking metric. Following the work by Kotov and Zhai (15), we focus on hard queries, which are defined as those with Precision at 10 (P@10) performance of 0 when only using the original query terms. We then evaluate how the performance can be improved by expanding the query with terms that are selected using a given strategy. There are 44 examples of such hard queries among the original 250 queries.

To expand queries based on ConceptNet, we first find all paths, consisting of up to 4 nodes, which connect any two of the query terms. For example, if the query is "drug crime organization", we find paths between the pairs (drug, crime), (drug, organization), and (crime, organization). We then add the words from the most natural paths to the original query. In particular, we add a fixed number of words by processing the paths in the order defined by the chosen path ranking method. To this end, we have considered the following strategies for ranking paths:

(1) naturalness score, which is calculated from our model trained on the open-domain (3rd) setting;
(2) pairwise score, which is the pairwise baseline;
(3) length score, which is the length baseline (i.e. shorter paths are better), with random tie-breaking;
(4) random score, which assigns a random score to each path;
(5) naturalness+length score, which first favors shorter paths, but uses our naturalness score to rank paths with same length.

To put our results into context, we also compared this method with a query expansion strategy based on the GloVe word embedding. Specifically, in this case we add the words that are closest to the query words, ordered by their cosine similarity in the vector space. Note that each of the considered strategies will yield an ordered list of possible words to add to the query. We study the retrieval performance of using different cut-offs in this list.

The P@10 and MAP performances are shown in Figure 3. We can see that for both evaluation metrics, the naturalness+length score achieves the highest performance, suggesting that while nodes in shorter paths are generally more relevant to original queries, considerable gains can be made by using naturalness to rank paths of the same length. The word embedding model also performs well, but it cannot reach the best overall performance of the two naturalness based methods.

4.5.3 Analogy Inference. Boteanu and Chernova (4) used relation paths to solve analogy questions of the following form:

```
Question: Dog:Animal::
       A: Table:Chair
       B: Apple:Fruit
       C: Fast:Slow
       D: Ice:Cold
```

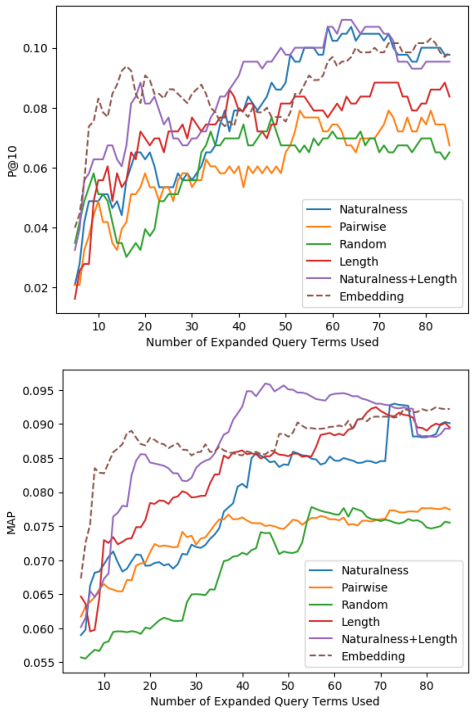


Figure 3: Performance of query expansion for hard queries in terms of Precision at 10 (P@10, top) and Mean Average Precision (MAP, bottom).

The goal is to select the word pair which has the same relation as the query pair (e.g. the correct answer in the question above is *B* because dog is a type of animal, and apple is a type of fruit). In their proposed method, the quality of the analogy $a : b :: c : d$ (meaning a is to b as c is to d) is calculated using the proportion of overlapping relations on paths connecting (a, b) and (c, d) respectively. Boteanu and Chernova (4) considered paths of two (i.e. direct connection) and three (i.e. with an intermediate hop) nodes.

We found that when direct connections exist between words in the query pair, using only these direct connection (and not including 3-node paths) achieves the best performance. However, not all query pairs have direct connections, in which case we are required to use 3-node paths. For our dataset of 85 questions, 74 have associated direct edge connections. Thus, we study the effect of only considering the most natural paths, rather than all of them, on the remaining 11 questions. Figure 4 shows the performance of the prediction as a function of the proportion of 3-node paths used. Using our model to filter 3-node paths, the best performance is obtained when only the 30% most natural paths are considered. As a comparison, the pairwise baseline is not able to outperform a random selection.

Across all three evaluations, we consistently find that including all (or too many) paths leads to sub-optimal performance, due to the inclusion of noisy paths, and that our proposed method outperforms existing heuristics based on word embedding and path length.

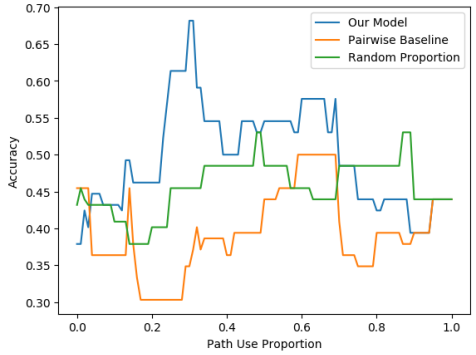


Figure 4: Performance on the 11 analogy questions for which direct connections are not available, using various proportions of natural paths.

5 CONCLUSIONS

Our work is motivated by the observation that commonsense knowledge graphs such as ConceptNet are noisy and informal, and that many of the relational paths in such knowledge graphs are therefore non-sensical. This means that the only way in which applications can successfully make use of relational paths from such resources is by relying on a method to filter out these non-sensical paths. While this problem has already been studied by others, previous solutions relied on relatively simple heuristics to select higher-quality paths, which we found to perform poorly in practice. In fact, in the extrinsic evaluation tasks, we found that some heuristics were not even able to outperform random selection (Figures 3 and 4).

Rather than engineering more sophisticated heuristics, the solution we proposed in this paper is to take a data-driven approach and learn the concept of path *naturalness* based on crowdsourced judgments. The main insights we obtained are as follows:

(1) although we intentionally left the interpretation of naturalness open to crowdsourcers, they do largely share a common perception of this concept;
(2) this concept can be effectively learned by an LSTM model, with accuracy close to an empirical upper bound derived from the inter-annotator agreement level; and
(3) our learned model outperforms previously proposed heuristics as a path selection method in several intrinsic and extrinsic evaluation tasks.

ACKNOWLEDGMENT

Steven Schockaert was supported by ERC Starting Grant 637277.

REFERENCES

[1] Lora Aroyo and Chris Welty. 2014. The Three Sides of CrowdTruth. *Human Computation* 1 (2014), 31–34.
[2] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. *The Semantic Web* (2007), 722–735.
[3] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*. ACM, 1247–1250.

[4] Adrian Boteanu and Sonia Chernova. 2015. Solving and Explaining Analogy Questions Using Semantic Networks. In *AAAI Conference on Artificial Intelligence*. AAAI, 1460–1466.

[5] Zied Bouraoui, Shoaib Jameel, and Steven Schockaert. 2018. Relation Induction in Word Embeddings Revisited. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING)*. 1627–1637.

[6] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. 2010. Toward an Architecture for Never-Ending Language Learning. In *AAAI Conference on Artificial Intelligence*. 1306–1313.

[7] Junpeng Chen and Juan Liu. 2011. Combining ConceptNet and WordNet for Word Sense Disambiguation. In *International Joint Conference on Natural Language Processing (IJCNLP)*. 686–694.

[8] Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. 2017. Chains of Reasoning over Entities, Relations, and Text Using Recurrent Neural Networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Vol. 1. 132–141.

[9] Georgiana Dinu, Angeliki Lazaridou, and Marco Baroni. 2015. Improving Zero-Shot Learning by Mitigating the Hubness Problem. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

[10] Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting Word Vectors to Semantic Lexicons. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL:HLT)*. Association for Computational Linguistics, 1606–1615.

[11] Matt Gardner, Partha Talukdar, Jayant Krishnamurthy, and Tom Mitchell. 2014. Incorporating Vector Space Similarity in Random Walk Inference over Knowledge Bases. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 397–406.

[12] Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing Knowledge Graphs in Vector Space. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[14] Diederik Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the International Conference Learning Representations (ICLR)*.

[15] Alexander Kotov and Chengiang Zhai. 2012. Tapping into Knowledge Base for Concept Feedback: Leveraging ConceptNet to Improve Search Results for Difficult Queries. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 403–412.

[16] Alicia Krebs, Alessandro Lenci, and Denis Paperno. 2018. SemEval-2018 Task 10: Capturing Discriminative Attributes. In *Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval)*. 732–740.

[17] Ni Lao and William W Cohen. 2010. Relational Retrieval Using a Combination of Path-Constrained Random Walks. *Machine Learning* 81, 1 (2010), 53–67.

[18] Douglas B Lenat. 1995. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM (CACM)* 38, 11 (1995), 33–38.

[19] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015. Modeling Relation Paths for Representation Learning of Knowledge Bases. In *Proceedings of the Conference Empirical Methods in Natural Language Processing (EMNLP)*. 705–714.

[20] Hugo Liu and Push Singh. 2004. ConceptNet—a Practical Commonsense Reasoning Tool-Kit. *BT Technology Journal* 22, 4 (2004), 211–226.

[21] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Advances in Neural Information Processing Systems (NIPS)*. 3111–3119.

[22] George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. 1990. Introduction to WordNet: An On-line Lexical Database. *International Journal Lexicography* 3, 4 (1990), 235–244.

[23] Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. 2015. Compositional Vector Space Models for Knowledge Base Completion. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL & IJCNLP)*. Association for Computational Linguistics, 156–166.

[24] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543.

[25] Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. 2010. Hubs in Space: Popular Nearest Neighbors in High-Dimensional Data. *Journal of Machine Learning Research (JMLR)* 11 (2010), 2487–2531.

[26] Push Singh, Thomas Lin, Erik T Mueller, Grace Lim, Travell Perkins, and Wan Li Zhu. 2002. Open Mind Common Sense: Knowledge Acquisition from the General Public. In *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*. Springer, 1223–1237.

[27] Robert Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In *AAAI Conference on Artificial Intelligence*. AAAI, 4444–4451.

[28] Kristina Toutanova, Victoria Lin, Wen-tau Yih, Hoifung Poon, and Chris Quirk. 2016. Compositional Learning of Embeddings for Relation Paths in Knowledge Base and Text. In *Procedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, Vol. 1. 1434–1444.

[29] Luis Von Ahn, Mihir Kedia, and Manuel Blum. 2006. Verbosity: A Game for Collecting Common-Sense Facts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*. ACM, 75–78.

[30] Ellen M Voorhees. 2004. Overview of the TREC 2004 Robust Retrieval Track. In *TREC*.

[31] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A Free Collaborative Knowledgebase. *Communications of the ACM (CACM)* 57, 10 (2014), 78–85.

[32] Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 564–573.

[33] Chang Xu, Yalong Bai, Jiang Bian, Bin Gao, Gang Wang, Xiaoguang Liu, and Tie-Yan Liu. 2014. RC-NET: A General Framework for Incorporating Knowledge into Word Representations. In *Procedings of the 23rd ACM International Conference Information and Knowledge Management (CIKM)*. ACM, 1219–1228.

[34] Jinxi Xu and W. Bruce Croft. 1996. Query Expansion Using Local and Global Document Analysis. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, USA, 4–11.

[35] George Kingsley Zipf. 1935. The Psycho-Biology of Language. (1935).