Neural Variational Entity Set Expansion for Automatically Populated Knowledge Graphs

Pushpendre Rastogi Adam Poliak Vince Lyzinski Benjamin Van Durme

Johns Hopkins University
pushpendre@jhu.edu

Abstract

We propose Neural Variational Set Expansion to extract actionable information from a noisy knowledge graph (KG) and propose a general approach for increasing the interpretability of recommendation systems. We demonstrate the usefulness of applying a variational autoencoder to the Entity Set Expansion task based on a realistic automatically generated KG.

1 Introduction

Imagine a physician trying to pin-point a specific diagnosis or a journalist investigating abuses of governmental power. In both scenarios, a *domain expert* may try to find answers based on prior known, relevant entities – either a list of diagnoses of with similar symptoms that a patient is experiencing or a list of known conspirators. Instead of manually looking for connections between potential answers and prior knowledge, a *searcher* would like to rely on an automatic *Recommender* to find the connections and answers for them, i.e. related entities.

In the information retrieval (IR) community, Entity Set Expansion (ESE) is the established task of recommending entities that are similar to a provided seed of entities.[1] ESE has been applied in Question Answering [44], Relation Extraction [18] and Information Extraction [13] settings. The physician and journalist in our example can not fully take advantage of IR advances in ESE for two main reasons. Recent advances 1) often assume access to a clean, large Knowledge Graph and 2) are uninterpretable.

Many advanced ESE algorithms rely on manually curated, clean Knowledge Graphs (KG), e.g. DBpedia [3] and Freebase [6]. In real-world settings, users rarely have access to clean KGs, and instead may rely on automatically generated KGs. Such KGs are often *noisy* because they are created from complicated and error-prone NLP processes – illustrated in Figure 1. For example, automatic KGs may include duplicate entities, associations (relations) between entities may be missing, and entities with similar names may be incorrectly disambiguated. These imperfections prevent machine learning approaches from performing well on automatically generated KGs. Furthermore, many ESE algorithms degrade as the sparsity and unreliability of KGs increases [31, 32]. Advanced ESE methods, especially those that rely on neural networks, are uninterpretable [26]. If a physician can not explain decisions, patients may not trust her and if a journalist can not demonstrate how a certain individual is acting unethically or above the law, a resulting article may lack credibility. Furthermore, uniterpretability may limit the applications of advancements in IR, and more broadly artificial intelligence, as humans "won't trust an A.I. unless it can explain itself."[2]

We introduce Neural Variational Set Expansion (NVSE) to advance the applicability of ESE research. NVSE is an unsupervised model based on Variational Autoencoders (VAEs) that receives a query, uses a Bayesian

In: Joint Proceedings of the First International Workshop on Professional Search (ProfS2018); the Second Workshop on Knowledge Graphs and Semantics for Text Retrieval, Analysis, and Understanding (KG4IR); and the International Workshop on Data Search (DATA:SEARCH18). Co-located with SIGIR 2018, Ann Arbor, Michigan, USA – 12 July 2018, published at http://ceur-ws.org

[1] We refer to the items in the seed as entities but they can also be referred to as items or elements

[2] https://nyti.ms/2hR1S15

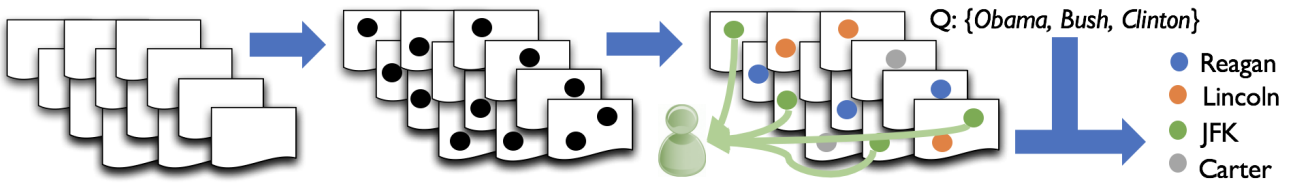


Figure 1: Our *Entity Set Expansion* (ESE) system assumes a corpus that has been labeled with entity mentions which are clustered via cross-document co-reference and linking to a knowledge base; together known as *entity discovery and linking* (EDL). Given a query containing *Obama*, *Bush*, and *Clinton*, the ESE system returns other U.S. presidents found in the KG.

approach to determine a latent concept that unifies entities in the query, and returns a ranked list of similar entities based on the previously determined unified latent concept. NVSE does not require supervised examples of queries and responses, nor pre-built clusters of entities. Instead, our method only requires sentences with linked entity mentions, i.e. spans of token associated with a KG entity, often included in automatically generated KGs.

NVSE is robust to noisy automatically generated KGs, thus removing the need to rely on manually curated, clean KGs. We evaluate NVSE on the ESE task using Tinkerbell [1], an automatically generated KG that placed first in the TAC KGP shared task. Unlike how ESE has been used to improve entity linking for KG construction [10], our goal is the opposite: we leverage noisy automatically generated KGs to perform ESE. NVSE is interpretable; it outputs **query rationales** – a summarization of features our models associates with the query – and **result justifications** – an ordered list of sentences from the underlying corpus that justify why our method returned that entity. Query rationales and result justifications are reminiscent of *annotator rationales* [46].

To our knowledge this is the first unsupervised neural approach for ESE as opposed to neural methods for supervised collaborative filtering [19]. All code and data is available at `github.com/se4u/nvse` and a video demonstration of the system is available at `youtu.be/sGO_wvuPIzM`.

2 Related Work

Methods dependent on external information. Since automatically generated KGs can be noisy, some methods utilize information beyond entity links and mentions to aid ESE. [28] use search engine query logs to extract attributes related to entities and [29] extract sets of instances associated with class labels based on web documents and queries. [27] use a large amount of web data as they apply a learned word similarity matrix extracted from a 200 billion word Internet crawl to the ESE task. Both [12]'s SEISA system and [39]'s Google Sets use lists of items from the Internet and try to determine which elements in the lists are most relevant to a query. [36] rely on given topic information about the queried entities to train a discriminative system. More recent approaches also use external information. [45] use LDA [5] to create word clusters for supervision, and [40] use manual annotations by Twitter users. [48] uses inter-entity links in knowledge graphs which are very sparse in automatically generated KGs [31, 32]. All of these approaches use more information than just entity links and mentions.

Methods for comparing entities. Set Expander for Any Language (SEAL) [41] and its variants [42, 43] learn similarities between new words and example words using methods like Random Walks and Random Walks With Restart. Similar to [21]'s using cosine and Jaccard similarity to find similar words, SEISA uses these metrics to expand sets. These methods are limited to only extracting words that coocur. Because they are applied on web-scale data, SEAL and SEISA assume entities will eventually coocur. This assumption might not be valid in an underlying corpus used to automatically generate a KG. In contrast to those approaches, NVSE finds similar entities based on a kernel between distributions.

Queries as natural language. In the INEX-XER shared task, queries were represented as natural language questions [8]. [23] and [47] propose methods to extract related entities in a KG based on a natural language query. This scenario is similar to a person interacting with a system like Amazon Alexa. However, our setup better reflects users searching for similar entities in a KG as it is more efficient for users to type entities of interest instead of natural language text.

Neural Collaborative Filtering. We are not the first to incorporate neural methods in a recommendation system. Recently, [11] and [19] presented deep auto-encoders for collaborative filtering. Collaborative Filtering assumes a large dataset of previous user interactions with the search engine. For many domains it is not possible

to create such a dataset since new data is added everyday and queries change rapidly based on different users and domains. Therefore, we propose the first neural method which does not use supervision for Entity Set Expansion.

3 Notation

Let $\mathcal{D}$ be the corpus of documents and $\mathcal{V}$ be the vocabulary of tokens that appear in $\mathcal{D}$. We define a document as a sequence of sentences and we define a sentence as a sequence of tokens. Let $\mathcal{X}$ be the set of entities discovered in $\mathcal{D}$ and we refer to its size as X. Each entity $x \in \mathcal{X}$ is linked to the tokens that mention x.[3] Let $\mathcal{V}'$ be the set of tokens linked to any $x \in \mathcal{X}$, and let $\mathcal{M}_x$ be the multiset of sentences that mention x in the corpus. For example, consider an entity named "Batman" and a document containing three sentences {Batman is good., He is smart. Life is good.}. "Batman" is linked to tokens Batman and He, $\mathcal{V}' = \{\text{Batman}, \text{He}\}$, and $\mathcal{M}_{\text{Batman}} = \{\text{Batman is good., He is smart.}\}$.

In ESE, a system receives query $\mathcal{Q}$ – a subset of $\mathcal{X}$ – and has to sort the elements remaining in $\mathcal{R} = \mathcal{X} \setminus \mathcal{Q}$. The elements that are most similar to $\mathcal{Q}$ should appear higher in the sorted order and elements dissimilar to $\mathcal{Q}$ should be ranked lower.

4 Baseline Methods

Before introducing NVSE, we describe the four baselines systems: BM25, Bayesian Sets, Word2Vecf and SetExpan. We do not compare to DeepSets [45], as it is a supervised method that requires entity clusters.

For each x, we create a feature vector $f_x \in \mathbb{Z}^{\text{F}}$ from $\mathcal{M}_x$, by concatenating three vectors that count how many times 1) a token in V appeared in $\mathcal{M}_x$ 2) a document in $\mathcal{D}$ mentioned x and 3) a token in $\mathcal{V}'$ appeared in $\mathcal{M}_x$. Thus, $\text{F} = \text{V} + \text{D} + \text{V}'$.

4.1 BM25

Best Match 25 (BM25) is "one of the most successful text-retrieval algorithms" [35]. [4] BM25 ranks remaining entities in $\mathcal{R}$ according to the score function

$$\underset{BM}{\text{score}}(\mathcal{Q}, x) = \sum_{i=1}^{\text{F}} \frac{\text{IDF}[i] f_x[i] \bar{f}_{\mathcal{Q}}[i](k_1+1)}{f_x[i] + k_1(1 - b + b\sum_j f_x[j]/\bar{L})},$$

where $f_x[j]$ denotes the j-th feature value in f_x, $\bar{f}_{\mathcal{Q}}$ is the sum of $f_x \forall x \in \mathcal{Q}$ and $\mathbb{I}$ is the indicator function. k_1 and b are hyperparameters that commonly set to 1.5 and 0.75 [22]. $\bar{L}$ is the average total count of a feature in the entire corpus and $\text{IDF}[i]$ is the inverse document frequency of the i^{th} feature (Appendix A).

4.2 Bayesian Sets

[9] introduced the Bayesian Sets (BS) method which converts ESE into a bayesian model selection problem. BS compares the probabilities that the query entities are generated from a single sample of a latent variable $z \in \Delta^{\text{F}}$ with the probability that the entities were generated from independent samples. Δ^{F} is the $\text{F}-1$ dimensional probability simplex. Note that z has the same dimensionality as the observed features. Given $\mathcal{Q}$ and π, the prior distribution of z, BS infers the posterior distribution of z, $p(z|\mathcal{Q})$, and computes the following score

$$\underset{BS}{\text{score}}(\mathcal{Q}, x) = \log \frac{E_{p(z|\mathcal{Q})}[p(x|z)]}{E_{\pi(z)}[p(x|z)]}. \tag{1}$$

[9] computed score_{BS} in close form by selecting the conditional probability, $p(x|z)$, from an exponential family distribution and setting π to be its conjugate prior. They showed that if $p(x|z)$ is multivariate Bernoulli then BS requires a single matrix multiplication (Appendix C) and we use this setting for our experiments.

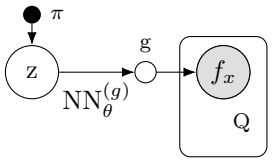
4.3 Word2Vecf

[20] generalize [25]'s Skip-Gram model as Word2Vecf to include arbitrary contexts. We embed entities with Word2Vecf by using the entity IDs as words [5] and the tokens in the sentences mentioning those entities as

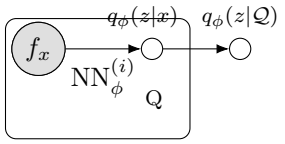
[3] We ignore confidence scores that entity linking systems often assign to a link because confidence scores will prevent us from using a multinomial distribution to model a document as a bag-of-words.

[4] Lucene replaced tf-idf with BM25 as its default algorithm: https://issues.apache.org/jira/browse/LUCENE-6789

[5] Converting entity mentions to entity IDs allows us to overcome issues related to embedding multi-word expressions as explained in [30].



(a) Generative Network



(b) Inference Network

Figure 2: The generative model of query generation is on the left and the variational inference network is on the right. Small nodes denote probability distributions, gray nodes are observations and the black node π is the known prior. $\text{NN}_\theta^{(g)}$ transforms z to g and the $\text{NN}_\phi^{(i)}$ transforms f_x to $q_\phi(z|x)$.

contexts. Note that all tokens in the sentence, except for some stop words, are used as contexts and not just co-occurrent entities. We rank the entities in the order of their total distance to the entities in the query set as

$$\underset{W2V}{\text{score}}(\mathcal{Q}, x) = -\sum_{\tilde{x} \in \mathcal{Q}} (v_x - v_{\tilde{x}})^2. \tag{2}$$

Here, v_x represents the L2-normalized embedding for x.

4.4 SetExpan

[37] introduce SetExpan, a SOTA framework combining context feature selection with ranking ensembles, for set expansion. SetExpan outperformed other SE methods such as SEISA in their evaluation. SetExpan represents entities by the contexts that they are mentioned in. For example, the context features for Batman from § 3 will be {__ is good, __ is smart}. The contexts are used to create a large feature vector which can be used to compute the inter-entity similarity. The authors argue that using all possible features for computing entity similarity can lead to overfitting and semantic drift. To combat these problems SetExpan builds the entity set iteratively by cycling between a context feature selection step and an entity selection step. In context feature selection, each context feature is assigned a score based on the set of currently expanded entities. Based on these scores, the context-features are reranked and the top few context features are selected. The entity selection proceeds by bootstrap sampling of the chosen context features and using those features to create multiple different ranked lists of entities. Multiple different ranked lists are finally combined via a heuristic method for ensembling different ranked lists to create a new set of expanded entities. This process is repeated to convergence to get the final list of expanded entities.

5 Neural Variational Set Expansion

Like BS, Neural Variational Set Expansion first determines the underlying concept, or topic, underlying the query and then ranks entities based on that concept. Our method differs from BS because we use a deep generative model with a low dimensional concept representation, to simulate how a concept may generate a query. Also we use a "distance" (§ 5.2) between posterior distributions for ranking entities in lieu of bayesian model comparison.

5.1 Inference Step 1: Concept Discovery

Our model (Fig. 2) is as follows: $z \in \mathbb{R}^d$ is a low dimensional latent gaussian random variable representing the concept of a query. z is sampled from a fixed prior distribution $\pi = \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$, i.e. $z \sim \pi$. The members of $\mathcal{Q}$ are sampled conditionally independently given z. z is mapped via a multi layer perceptron (MLP), called $\text{NN}_\theta^{(g)}$, to g, the p.m.f. of a multinomial distribution that generates f_x, the features of x. $\text{NN}_\theta^{(g)}$ is a neural network with a softmax output layer and parameters θ. $f_x \in \mathbb{Z}^{\text{F}}$ are sampled i.i.d. from $p(f|z, \theta) = \text{NN}_\theta^{(g)}(z)$.[6]

In other words, the vector f_x contains the counts of observed features for x that were sampled from g, and g was itself sampled by passing a gaussian random variable through a neural network.

Under this deep-generative model a concept vector can simultaneously trigger multiple observed features. This allows us to capture the correlations amongst features triggered by a concept. For example, the concept of `president` can simultaneously trigger features such as white house, executive order, or airforce one.

[6] Our generative model is inspired by [24]'s NVDM. They assume that a single latent variable generates only one observation, but we posit that the same latent variable z generates all observations in $\mathcal{Q}$.

In order to infer the latent variable z ideally we should compute $p_\theta(z|\mathcal{Q})$, the posterior distribution of z given the observations $\mathcal{Q}$. Unfortunately, this computation is intractable because the prior is not conjugate to the likelihood that has a neural network. Another problem is that it is unrealistic to assume access to a large set of ESE queries at training time, because user's information needs keep changing, therefore the approach used by [45] in DeepSets to discriminatively learn a neural scoring function is *impractical* for set expansion. For the same reason it is also not possible to learn a single neural network whose input is $\mathcal{Q}$ and which directly approximates $p_\theta(z|\mathcal{Q})$. Therefore it is non-trivial to apply the VAE framework to ESE. To overcomes these problems we make the assumption that a query $\mathcal{Q}$ is conjunctive in nature, i.e. if entity x_1 and x_2 are present in $\mathcal{Q}$ then results that are relevant to *both* x_1 and x_2 simultaneously should be given a higher ranking than results that are related to x_1 but not x_2 or vice-versa. We implement the conjunction of entities in a query by combining the *Product of Experts* [14] approach with the *Variational Autoencoder* (VAE) [16] method to approximate $p_\theta(z|\mathcal{Q})$.

We first map each x to an approximate posterior $q_\phi(z|x)$ via a neural network $\text{NN}_\phi^{(i)}$ and then we take their product to approximate $p_\theta(z|\mathcal{Q})$.

$$p_\theta(z|\mathcal{Q}) \approx q_\phi(z|\mathcal{Q}) \propto \prod_{x \in \mathcal{Q}} q_\phi(z|x).$$

The ϕ parameters are estimated by minimizing $KL(q(z|x) \;||\; p(z|x))$ as shown in § 5.3.[7] The benefit of the POE approximation is that the posterior approximation $q_\phi(.|x)$ for each entity x in $\mathcal{Q}$ acts as an expert and the product of these experts will assign a high value to only that region where all the posteriors assign a high value. Therefore the POE approximation is a way of implementing conjunctive semantics for a query. Another benefit is that if $q_\phi(.|x)$ is an exponential family distribution with a constant base measure whose natural parameters are the output of $\text{NN}_\phi^{(i)}$, then the product of the distributions $\prod_x q_\phi(\cdot|x)$ lies in the same exponential family whose natural parameters are simply the sum of individual neural network outputs.[8,9] We use $\text{NN}_\phi^{(i)}$ to compute the mean and log-variance of the gaussian distribution $q_\phi(z|x)$ (3) that we convert to the natural parameters of a Gaussian (4). Next, we add the natural parameters of the individual variational approximations ξ_x, Γ_x to compute the parameters $\xi_\mathcal{Q}, \Gamma_\mathcal{Q}$ for $q_\phi(z|\mathcal{Q})$ (5). Finally, we compute $q_\phi(z|\mathcal{Q})$ (6).

$$\mu_x, \Sigma_x = \text{NN}_\phi^{(i)}(f_x) \tag{3}$$

$$\xi_x, \quad \Gamma_x = \mu_x \Sigma_x^{-1}, \quad \Sigma_x^{-1}. \tag{4}$$

$$\xi_\mathcal{Q}, \quad \Gamma_\mathcal{Q} = \textstyle\sum_{x \in \mathcal{Q}} \xi_x, \quad \sum_{x \in \mathcal{Q}} \Gamma_x. \tag{5}$$

$$q_\phi(z|\mathcal{Q}) = \mathcal{N}_c(z|\xi_\mathcal{Q}, \Gamma_\mathcal{Q}) \tag{6}$$

$\mathcal{N}_c(z|\xi, \Gamma)$ is the multi-variate Gaussian distribution in terms of its natural parameters –

$$\frac{|\Gamma|^{1/2}}{(2\pi)^{D/2}} \exp\left(-\frac{(z^T \Gamma z - 2\xi^T z + \xi^T \Gamma^{-1} \xi)}{2}\right).$$

5.2 Inference Step 2: Entity Ranking

In order to rank the entities $x \in \mathcal{R}$, we design a similarity score between the probability distributions $q_\phi(z|\mathcal{Q})$ and $q_\phi(z|x)$ as an efficient substitute for bayesian model comparison. We use the distance between precision weighted means $\xi_\mathcal{Q}$ and ξ_x to define our "distance" function as

$$\underset{NVSE}{\text{score}}(\mathcal{Q}, x) = -||\xi_\mathcal{Q} - \xi_x||^2. \tag{7}$$

Our inter-distribution "distance" is not a proper distance because it changes as the location of both the input distributions is shifted by the same amount. We experimented with more standard, reparameterization invariant, divergences and kernels such as the KL-divergence and the Probability Product Kernel [15], see (Appendix D), but we found our approach to be faster and more accurate. We believe this is because the regularization from the prior that encourages the posteriors to be close to the origin makes shift invariance unnecessary.

[7] This is a generalization of [7] combining variational approximations of posterior distributions since the product of gaussians is a Gaussian distribution.

[8] Also notice that the POE approach recommends adding the *outputs* of the neural networks which is different than concatenating the features for all x in $\mathcal{Q}$ or naively adding the *inputs* of the neural network. (Appendix B) gives more details.

[9] Recently, [45] gave a theorem that any permutation invariant function of sets must be representable as the function of a sum of features of elements of the set. We note that our POE approximation also has a similar form and is permutation invariant.

5.3 Unsupervised Training

NVSE is trained in an unsupervised fashion to learn its parameters θ and ϕ. [16, 34] proposed the VAE framework for learning richly parameterized conditional distributions $p_\theta(x|z)$ from unlabeled data. We follow [16]'s reparameterization trick to train a VAE and maximize the *Evidence Lower Bound*:

$$E_{q_\phi(z|x)}[\log p_\theta(x|z)] - KL(q_\phi(z|x)||p(z)). \tag{8}$$

During training, we do not have access to any clustering information or side information that tells us which entities can be grouped together. Therefore we assume that the entities $x \in \mathcal{X}$ were generated i.i.d. The model during training looks the same as Figure 2 but with one difference: Q is a singleton set of just one entity.[10] Note that our learning method requires no supervision in contrast to methods like Deep Sets which require cluster information, or Neural Collaborative filtering methods which require a large dataset of user interactions.

6 Interpretability

We introduce a general approach for interpreting ESE models based on *query rationales* to explain the latent concept the model discovered and *result justifications* to provide evidence for why the system ranked an entity highly. Based on query rationales and result justifications, users can add weights to entities in a query to tell the system what aspects of the query to focus on or ignore.

6.1 Query Rationale

A *Query Rationale* is a visualization of the latent beliefs of the ESE system given the query $\mathcal{Q}$. Given $\mathcal{Q}$, we construct a feature-importance-map $\gamma_\mathcal{Q}$ that measures the relative importance of the features in f_x and we show the top features according to $\gamma_\mathcal{Q}$ as "Query Rationales". Recall that the j^{th} component of f_x, associated with entity x, measures how often the j^{th} feature co-occurred with x. We now present how we construct $\gamma_\mathcal{Q}$ for NVSE and the baselines.

For BM25, $\gamma_\mathcal{Q}$ is simply $\bar{f_\mathcal{Q}}$. In BS, $\gamma_\mathcal{Q}$ is the weights from (11b): for each j^{th} component of f_x,

$$\gamma_\mathcal{Q}[j] = \log \frac{\tilde{\alpha}_\mathcal{Q}[j]\beta[j]}{\alpha[j]\tilde{\beta}_\mathcal{Q}[j]}.$$

The benefit of generative methods such as BS and NVSE is that for them query rationales can be computed as a natural by-product of the generative process instead of as ad-hoc post-processing steps. For NVSE, ideally $\gamma_\mathcal{Q}$ should be the posterior distribution $p_\theta(f|\mathcal{Q})$. Since this is intractable we approximate it by sampling the inference network:

$$p_\theta(f|\mathcal{Q}) = E_{p_\theta(z|\mathcal{Q})}[p_\theta(f|z,\mathcal{Q})] \approx E_{q_\phi(z|\mathcal{Q})}[p_\theta(f|z)].$$

We further approximate the expectation with a single sample of the mean of $q_\phi(z|\mathcal{Q})$. Finally the feature importance map for NVSE is:

$$\gamma_\mathcal{Q} = p_\theta(f|E[q_\phi(z|\mathcal{Q})]).$$

Because Word2Vecf finds the nearest-neighbor between entity embeddings, which are produced through a complicated learning process operating on the whole text corpus, it does not provide a natural way to determine the importance of a single sentence and therefore it is not possible to say what was the effect of a particular sentence on the query results. Similarly, since the SetExpan method works by extracting context features and iteratively expanding this feature set, it is not possible to determine the effect of a single sentence on the final search results.

6.2 Result Justifications

We define result justifications as sentences in $\mathcal{M}_x$ that justify why an entity was ranked highly for a given query. Ranking the sentences that mention an entity is similar to ranking entities in $\mathcal{R}$. Just as we create a feature vector for each x, we create a feature vector for each sentence in $\mathcal{M}_x$ and use the same scoring function to rank the sentences based on the query. While computing a score for entity x based on a query, we also score each sentence in $\mathcal{M}_x$. Our approach to generate interpretable result justifications is agnostic to ESE methods with

[10] More informally, we remove the plates from Figure 2.

the caveat that for methods like Word2Vecf and SetExpan this will require retraining or reindexing over the corpus for each query. Our approach will not be feasible for such methods.

6.3 Weighted queries

Any recommendation system can occasionally fail to provide good results for a query. To improve a system's responses in such cases we enable users to guide NVSE's results by using entity weights to influence the posterior distribution over topics.

If a user provides weights $\boldsymbol{\tau} = \{\tau_x \mid x \in \mathcal{Q}\}$, we compute the query features as

$$\xi_{\mathcal{Q},\boldsymbol{\tau}}, \quad \Gamma_{\mathcal{Q},\boldsymbol{\tau}} = \sum_{x\in\mathcal{Q}} \tau_x \xi_x, \quad \sum_{x\in\mathcal{Q}} |\tau_x| \Gamma_x. \tag{9}$$

The above formulae have an intuitive explanation: when an entity has a higher weight then the precision over the concepts activated by that entity is increased according to the magnitude of the weight, and the value of the precision weighted mean is also weighted by the user supplied weights. In turn, an entity with zero weight has zero effect on the final search result and entities with a high negative weight return entities diametrically opposite to that entity with higher confidence.

Weights can be applied to other methods as well. BM25 can multiply each f_x by x's weights when computing $\bar{f}_{\mathcal{Q}}$, and Word2Vecf can use a weighted average. It is not straight-forward to incorporate weights in BS and SetExpan systems. One possible way is to use bootstrap resampling of the query entities according to a softmax distribution over entity weights, but bootstrapping makes the system non-deterministic and therefore even more opaque for a user. Also bootstrap resampling requires multiple query executions and it is not straight-forward to combine the outputs of different search queries; therefore we do not advocate for bootstrapping.

7 Comparative Experiments

We test the hypothesis that NVSE can help bridge the gap between advances in IR and real world use cases. We use human annotators on Amazon Mechanical Turk (AMT) to determine whether NVSE finds more relevant entities than our baseline methods in a real world, automatically generated KG.

7.1 Dataset

TinkerBell [1] is a KG construction system that achieved top performance in TAC-KGP2017 evaluation.[11] We used it as our automatic KG. For each entity e in TinkerBell we create $\mathcal{M}_e$ by concatenating all sentences that mention e and remove the top 100 most frequent features in the corpus from $\mathcal{M}_e$ to clean stop words. Tinkerbell was constructed from the TAC KGP 2017 evaluation source corpus, `LDC2017E25`, that contains 30K English documents and 60K Spanish and Chinese documents. [12] Half of the English documents come from online discussion forums and the other half from news sources, e.g. Reuters or the New York Times. Our experiments only use the 77,845 EDL entities within TinkerBell that are assigned the type `Person`. We use these links to create a map from DBPedia categories to entities in TinkerBell, say M. Each entity in TinkerBell is associated to spans of characters that mention that entity. We tokenize and sentence segment the documents in LDC2017E25 and associate sentences to each entity corresponding to mentions. In the end we get 344,735 sentences associated to the 77K entities. The median number of sentences associated to an entity is 1 and the maximum number of sentences is 4638 for the *Barack Obama* entity.[13] This is a good example of how automatic KGs differ from manually curated KGs. In TinkerBell most of the entities appear in only a single sentence so only a single fact may be known about them. In contrast KGs like FreeBase and DBPedia have a more uniform coverage of facts for entities present in them. Another difference is that relational information such as ancestry relations between entities are much more noisy in an automatically generated KB than in DBPedia which relies on manually curated information present in Wikipedia.

[11] Tinkerbell constructed a KG from `LDC2017E25` that contains $30K$ English documents. Half of them are from online forums and the other half from Reuters and NYT. We focused on the $77,845$ entities from English documents appearing in $344,735$ sentences. $25,149$ entities were also linked to DBPedia.

[12] `tac.nist.gov/2017/KGP/data.html`

[13] The mean is 4.43, the standard deviation is 29.19, the minimum number of sentences for an entity is 1, the maximum number of sentences is 4638, and the median is 1 (44,317 entities).

Category	Entities
(1 Sent./Ent.) American Jazz Singers	Paula West, Natalie Cole, Chaka Khan
(2-10 Sent.) Australian Major Golfers	Marc Leishman, David Graham, James Nitties
(11-100 Sent.) The Apprentice (U.S) Contestants	Maria, Rod Blagojevich, Dennis Rodman, Joan Rivers, Piers Morgan

Table 1: Examples of randomly created queries

7.2 Implementation Details

We prune the vocabulary by removing any tokens that occur less than 5 times across all entities. We end up with, F=105448, V = 61311, D = 24661, and V′ = 19476. We used BM25 implemented in Gensim [33] and we implemented BS ourselves. We choose $\lambda = 0.5$, out of 0, 0.5, or 1, after visual inspection. We used Word2Vecf and SetExpan codebases released by the authors.[14] For NVSE, we set d=50, σ=1. The generative network $\text{NN}_\theta^{(g)}$ does not have hidden layers and the inference network $\text{NN}_\phi^{(i)}$ has 1 hidden layer of size 500 with a tanh non-linearity and two output layers for the mean μ_x and log of the diagonal of the variance Σ_x. We use a diagonal Σ_x. [15] For Word2Vecf, we used $d = 100$ to use the same number of parameters per entity as in NVSE. We trained with default hyperparameters for 100 iterations. We used SetExpan with the default hyperparameters as well except that we limited the number of maximum iterations to 3 since we only needed top 4 entities for our experiments.

7.3 Experimental Design

Prior work typically evaluates ESE on a small number of queries, constituting the most frequent entities, e.g. [9] reported results for 10 queries with highly cited authors and [37] used 20 test queries created of 2000 most frequent entities in Wikipedia. However in automatic KGs, most entities are mentioned only a few times. For example 60% of the entities in TinkerBell are mentioned once. We are primarily interested in unbiased evaluation over such entities, therefore we stratified the evaluation queries into three types.

The 1st type contains entities mentioned in only 1 sentence, the 2nd contains entities appearing in $2-10$ sentences, and the 3rd contains entities mentioned in $11-100$ sentences. We also stratified queries based on whether they had 3, or 5 entities. For each query type we randomly generate 80 queries by first sampling 80 Wikipedia categories and then sampling entities from those categories that were also part of the TinkerBell KG. This results in 480 queries. See Table 1 for examples.

For each query, we showed the names and first paragraphs from the Wikipedia abstracts of the query's entities, to help the AMT workers disambiguate entities unfamiliar to them. Then we showed the workers the top 4 entities returned by each system. Each resultant entity was shown with up to 3 *justification* sentences.[16] Since SetExpan and Word2Vecf do not return justifications, we used NVSE to extract justifications for their results. We asked workers to rank the systems between 1, the best system, to 3, the worst; and we allowed for ties. The annotators found it difficult to compare results from 5 systems at a time so we split our evaluation into two groups. Group 1 compared NVSE to BS and BM25, and group 2 compared NVSE to SetExpan and Word2Vecf. We randomized the placement of the lists so that the workers could not figure out which system created which list.

7.4 Results

Table 2 shows the number of times the annotators ranked each system as the best out of the 80 queries. Over all queries, NVSE returned better results compared to the 4 baselines systems. It performed best with 5 entities in the query where each entity was only mentioned up to 10 times in the corpus. This shows that NVSE is able to discern better quality topics from multiple entities with sparse data. Extended results showing second and third place rankings of the systems are given in Table 5 in the appendix which show that in cases that when NVSE does not rank first it is typically chosen as the second ranking system.

[14] https://bitbucket.org/yoavgo/word2vecf, github.com/mickeystroller/SetExpan
[15] Training NVSE on 1 Tesla K80 using the Adam optimizer with learning rate $5e^{-5}$ and minibatch size 64 took 12 hours.
[16] Figure 3 in (Appendix E) illustrates the AMT interface.

Ents. In Query	Sents. Per Ent.	Group 1			Group 2		
		NVSE	BM25	BS	NVSE	SetEx	W2Vecf
3	1	27	**38**	15	**51**	14	15
	2-10	**29**	25	26	**49**	13	18
	11-100	**35**	23	22	**44**	10	26
5	1	**38**	25	17	**58**	19	3
	2-10	**40**	27	13	**53**	19	8
	11-100	24	**33**	24	**52**	11	17
	Total	**193**	171	117	**307**	86	87

Table 2: The number of times a system was ranked 1^{st} over 80 queries compared to other systems in the same group. Ties were allowed so some rows may not sum to 80. Bold highlights the system with the most 1^{st} in its group. Extended results with second and third place rankings of the system are shown in Table 5.

merger	procurement	husband	iii	best	very	game	tackle	wild	lighting
industry	securities	sister	house	its	most	drill	fuzzy	holly	costumes
premiers	AP-doc1	she	labor	good	end	offensive	21	exhibit	fashion
NYT-doc2	analyst	her	king	some	do	coach	doc	martins	nightclub
protection	founders	daughter	church	only	such	artur	doc3	thriller	theatrical
j=3, *business finance*		j=14, *family royalty*		j=20, *"qualifier"*		j=33, *football sports*		j=37, *entertainment movie*	

Table 3: The first row contains top 10 features most similar to z_j. The bottom row contains labels agreed upon by the authors; we loosely refer to the group where $j = 20$ as "qualifiers". Underscored words signify that the feature came from $\mathcal{V}'$.

The IR method BM25 was the strongest baseline, outperforming BS and SetExpan, and even NVSE in two settings. We believe that this is because of the low-resource conditions of our evaluation where ad-hoc IR methods can have an advantage. Another reason why BM25 worked very well in our evaluation was because of the lack of auxilliary signals such as entity inter-relations and entity links and because all the entities were of person type. This makes our task different from the entity list completion (ELC) task [4] and a bit simpler for methods that focus heavily on lexical overlap. Another difference between the ESE task and the ELC task was that in the ELC task a descriptive prompt describing the query was also given to the users while evaluating the relevance of the returned results whereas no such prompt was given in the ESE task. We also found that sometimes BM25 was rated highly because it returned results that were highly relevant to a single query entity instead of being topically similar to all entities. For example, on the query associated with "The Apprentice Contestants" BM25's results solely focused on Dennis Rodman, but NVSE tried to infer a common topic amongst entities and returned generic celebrities which annotators did not prefer.

On entities with little data, Word2Vecf and SetExpan perform poorly. Word2Vecf requires large amounts of data for learning useful representations [2] which explains why it performs poorly in our evaluation. The SetExpan algorithm directly uses context features extracted from the mentions of an entity, and returns entities with the same context features. This approach can overfit with low data. Even though SetExpan uses an ensembling method to reduce the variance of the algorithm, we believe using context-features causes overfitting when an entity appears in only a few sentences. Lastly, we believe that BS suffers because its impoverished generative model has neither non-linearities, nor low-dimensional topics for modeling correlations amongst tokens.

Abu Bakr Baghdadi (1)	Osama Bin Laden (1)	O.B. Laden (1.5) A.B. Baghdadi (1)	O.B. Laden (0.5) A.B. Baghdadi (2)	O.B. Laden (-0.2) A.B. Baghdadi (1)
qaida, iraq, abu, baghdadi, bakr, al, leader, attacks	bin, laden, osama, al, cia, pakistani, afridi, qaida	qaida, al, u, pakistani, cia, qaeda, government, killed	qaida, al, leader, attacks, u, killed, officials, islamic	bakr, baghdadi, abu, iraq, al, sectarian, nuri, kurdish

Table 4: The top row represents a query with weights in parentheses and the bottom row lists corresponding query rationales.

8 Analyzing Interpretability

We now attempt to understand the similarity relations encoded in NVSE's internal concept representations to understand what it is learning. We also provide examples of how query rationales and query weights can help

users fine-tune their queries.

8.1 Understanding the concept space

To gain some insight into the distribution over concepts inferred by NVSE we determined the top 10 words activated by individual dimension of z by computing $\text{NN}_\theta^{(g)}(e_j)$ where e_j is a one-hot vector in $\mathbb{R}^{50}$. Table 3 shows the top 10 words for selected components of z. We can easily recognize that dimensions $3, 33$ and 37 of z represent finance, sports, and entertainment. Even though we did not constrain z to be component-wise interpretable, this structure naturally emerged after training.

8.2 Weights & Query Rationale

Table 4 depicts how the *query rationale* returned by NVSE changes in response to entity weights. In the first column the query is {Abu Bakr Baghdadi} and the query rationale tells us that NVSE focuses on *iraq*, *baghdadi* etc. The second column shows a different query {Osama Bin Laden} and the query rationales changes accordingly to *pakistani* and *osama*. The third and fourth column show rationales when the weights on "Laden" and "Baghdadi" are varied. When more weight is put on "Laden" then the query rationales contain more features that are associated to him, and when more weight is put on "Baghdadi", then features such as "islamic" which is a token from his organization are returned. The last column shows an interesting configuration where a user is effectively asking for results that are similar to "Baghdadi" but dissimilar to "Laden" and the feature for *kurdish* gets activated. Since the system returns results in under 100ms, the user can fine-tune her query in real-time with the help of these query rationales.

We give one more example of the utility of negative weights: When $\mathcal{Q} = \{\text{Brady}\}$, NVSE's rationale is [*brady, game, patriots, left, knee, field, tackle*], indicating that NVSE associated the "Brady" entity with Tom Brady who is a member of the Patriots football team. When we added "Wes Welker" to $\mathcal{Q}$ with a negative weight, the query rationale changed to [*brady, game, left, tackle, knee, back, field*]. Since Wes is a Patriots receiver who received a negative weight in the query, NVSE deactivated the *patriots* feature and activated the *tackle* feature, opposite to a *receiver*.

9 Conclusion

We introduced NVSE as a step towards making advances in entity set expansion useful to real-world settings. NVSE is a novel unsupervised approach based on the VAE framework that discovers related entities from noisy knowledge graphs. NVSE ranks entities in a KG using an efficient and fast scoring function (7), ranking 80K entities on a commodity laptop in 100 milliseconds.

Our experiments demonstrated that NVSE can be applied in real-world settings where automatically generated KGs are noisy. NVSE outperformed state of the art ESE systems and other strong baselines on a real world KG. We also presented a flexible approach to interpret ESE methods and justify their recommendations.

In future work, we will extend our work by improving our model using more powerful auto-encoders such as the Ladder VAE [38], secondly we will experiment with the use of side information such as links from a KG through the use of Graph Convolutional Networks [17]. Third, we will like to quantitatively measure how query rationales and justifications help users in accomplishing their search task. Finally, we will incorporate confidence scores from the KG in our model. Although there may be future work to improve our ESE method, we believe that NVSE serves as a significant step towards utilizing KGs and semantics for information retrieval and understanding in real world settings.

References

[1] Al-Badrashiny, M., Bolton, J., Tejavsi Chaganty, A., Clark, K., Harman, C., Huang, L., Lamm, M., Lei, J., Lu, D., Pan, X., Paranjape, A., Pavlick, E., Peng, H., Qi, P., Rastogi, P., See, A., Sun, K., Thomas, M., Tsai, C.T., Wu, H., Zhang, B., Callison-Burch, C., Cardie, C., Ji, H., Manning, C., Muresan, S., C. Rambow, O., Roth, D., Sammons, M., Van Durme, B.: Tinkerbell: Cross-lingual cold-start knowledge base construction. In: Text Analysis Conference (TAC). (2017)

[2] Altszyler, E., Sigman, M., Slezak, D.F.: Comparative study of LSA vs word2vec embeddings in small corpora: a case study in dreams database. CoRR **abs/1610.01520** (2016)

[3] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: Dbpedia: A nucleus for a web of open data. The semantic web (2007) 722–735

[4] BALOG, K.: Overview of the trec 2009 entity track. Proc. TREC2009 (2009)

[5] Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine Learning research **3**(Jan) (2003) 993–1022

[6] Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: A collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, New York, NY, USA, ACM (2008)

[7] Bouchacourt, D., Tomioka, R., Nowozin, S.: Multi-level variational autoencoder: Learning disentangled representations from grouped observations. arXiv preprint arXiv:1705.08841 (2017)

[8] Demartini, G., Iofciu, T., De Vries, A.P.: Overview of the inex 2009 entity ranking track. In: Proceedings of the Focused Retrieval and Evaluation, and 8th International Conference on Initiative for the Evaluation of XML Retrieval. INEX'09, Berlin, Heidelberg, Springer-Verlag (2010) 254–264

[9] Ghahramani, Z., Heller, K.A.: Bayesian sets. In: Advances in neural information processing systems. (2006) 435–442

[10] Gottipati, S., Jiang, J.: Linking entities to a knowledge base with query expansion. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics (2011) 804–813

[11] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web. WWW '17, Republic and Canton of Geneva, Switzerland, International World Wide Web Conferences Steering Committee (2017) 173–182

[12] He, Y., Xin, D.: Seisa: set expansion by iterative similarity aggregation. In: Proceedings of the 20th international conference on World wide web, ACM (2011) 427–436

[13] He, Y., Grishman, R.: Ice: Rapid information extraction customization for nlp novices. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, Denver, Colorado, Association for Computational Linguistics (June 2015) 31–35

[14] Hinton, G.E.: Products of experts. In: 1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470). Volume 1. (1999) 1–6 vol.1

[15] Jebara, T., Kondor, R., Howard, A.: Probability product kernels. Journal of Machine Learning Research **5**(Jul) (2004) 819–844

[16] Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)

[17] Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: Proceedings of ICLR. (2017)

[18] Lang, J., Henderson, J.: Graph-based seed set expansion for relation extraction using random walk hitting times. In: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Atlanta, Georgia, Association for Computational Linguistics (June 2013) 772–776

[19] Lee, W., Song, K., Moon, I.C.: Augmented variational autoencoders for collaborative filtering with auxiliary information. In: ACM Conference on Information and Knowledge Management. Number 6, doi: 10.475/123 4, ACM (Nov 2017)

[20] Levy, O., Goldberg, Y.: Dependency-based word embeddings. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Baltimore, Maryland, Association for Computational Linguistics (June 2014) 302–308

[21] Lin, D.: Automatic retrieval and clustering of similar words. In: Proceedings of the 17th international conference on Computational linguistics-Volume 2, Association for Computational Linguistics (1998) 768–774

[22] Manning, C.D., Raghavan, P., Schtze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)

[23] Metzger, S., Schenkel, R., Sydow, M.: Aspect-based similar entity search in semantic knowledge graphs with diversity-awareness and relaxation. In: Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014 IEEE/WIC/ACM International Joint Conferences on. Volume 1., IEEE (2014) 60–69

[24] Miao, Y., Yu, L., Blunsom, P.: Neural variational inference for text processing. In: International Conference on Machine Learning. (2016) 1727–1736

[25] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. (2013) 3111–3119

[26] Mitra, B., Craswell, N.: Neural Models for Information Retrieval. ArXiv e-prints (May 2017)

[27] Pantel, P., Crestan, E., Borkovsky, A., Popescu, A.M., Vyas, V.: Web-scale distributional similarity and entity set expansion. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Singapore, Association for Computational Linguistics (August 2009) 938–947

[28] Paşca, M., Van Durme, B.: What you seek is what you get: Extraction of class attributes from query logs. In: IJCAI. (2007)

[29] Paşca, M., Van Durme, B.: Weakly-supervised acquisition of open-domain classes and class attributes from web documents and query logs. Proceedings of ACL-08: HLT (2008) 19–27

[30] Poliak, A., Rastogi, P., Martin, M.P., Van Durme, B.: Efficient, compositional, order-sensitive n-gram embeddings. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, Valencia, Spain, Association for Computational Linguistics (April 2017) 503–508

[31] Pujara, J., Augustine, E., Getoor, L.: Sparsity and noise: Where knowledge graph embeddings fall short. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, Association for Computational Linguistics (September 2017) 1751–1756

[32] Rastogi, P., Lyzinski, V., Van Durme, B.: Vertex nomination on the cold start knowledge graph. Technical report, Human Language Technology Center of Excellence (2017)

[33] Řehůřek, R., Sojka, P.: Software Framework for Topic Modelling with Large Corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, Valletta, Malta, ELRA (May 2010) 45–50

[34] Rezende, D.J., Mohamed, S., Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. arXiv preprint arXiv:1401.4082 (2014)

[35] Robertson, S., Zaragoza, H.: The probabilistic relevance framework: Bm25 and beyond. Found. Trends Inf. Retr. **3**(4) (April 2009) 333–389

[36] Sadamitsu, K., Saito, K., Imamura, K., Kikui, G.: Entity set expansion using topic information. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, Oregon, USA, Association for Computational Linguistics (June 2011) 726–731

[37] Shen, J., Wu, Z., Lei, D., Shang, J., Ren, X., Han, J.: Setexpan: Corpus-based set expansion via context feature selection and rank ensemble. In Ceci, M., Hollmén, J., Todorovski, L., Vens, C., Džeroski, S., eds.: Machine Learning and Knowledge Discovery in Databases, Cham, Springer International Publishing (2017) 288–304

[38] Sønderby, C.K., Raiko, T., Maaløe, L., Sønderby, S.K., Winther, O.: Ladder variational autoencoders. In: Advances in neural information processing systems. (2016) 3738–3746

[39] Tong, S., Dean, J.: System and methods for automatically creating lists (March 25 2008) US Patent 7,350,187.

[40] Vartak, M., Thiagarajan, A., Miranda, C., Bratman, J., Larochelle, H.: A meta-learning perspective on cold-start recommendations for items. In Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., eds.: Advances in Neural Information Processing Systems 30. Curran Associates, Inc. (2017) 6907–6917

[41] Wang, R.C., Cohen, W.W.: Language-independent set expansion of named entities using the web. In: Seventh IEEE International Conference on Data Mining (ICDM 2007). (Oct 2007) 342–350

[42] Wang, R.C., Cohen, W.W.: Iterative set expansion of named entities using the web. In: 2008 Eighth IEEE International Conference on Data Mining. (Dec 2008) 1091–1096

[43] Wang, R.C., Cohen, W.W.: Automatic set instance extraction using the web. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1, Association for Computational Linguistics (2009) 441–449

[44] Wang, R.C., Schlaefer, N., Cohen, W.W., Nyberg, E.: Automatic set expansion for list question answering. In: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, Honolulu, Hawaii, Association for Computational Linguistics (October 2008) 947–954

[45] Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R.R., Smola, A.J.: Deep sets. In Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., eds.: Advances in Neural Information Processing Systems 30. Curran Associates, Inc. (2017) 3394–3404

[46] Zaidan, O., Eisner, J., Piatko, C.: Using annotator rationales to improve machine learning for text categorization. In: Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference. (2007) 260–267

[47] Zhang, X., Chen, Y., Chen, J., Du, X., Wang, K., Wen, J.R.: Entity set expansion via knowledge graphs. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR '17, New York, NY, USA, ACM (2017) 1101–1104

[48] Zheng, Y., Shi, C., Cao, X., Li, X., Wu, B.: Entity set expansion with meta path in knowledge graph. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer (2017) 317–329

A IDF computed for BM25

BM25 is computed based on the average total count of a feature in the entire corpus and IDF$[i]$ is the inverse document frequency of the i^{th} feature amongst all documents, which is defined as

$$\begin{aligned}\text{IDF}[i] =& \log \frac{\text{X} - \text{DF}[i] + 0.5}{\text{DF}[i] + 0.5} \\ \text{DF}[i] =& \sum_{x \in \mathcal{X}} \mathbb{I}[f_x[i] > 0].\end{aligned}$$

B Computing Product of Experts for Deep-Exponential Families

In this section we show how the product of experts can be computed simply by adding the output of the neural networks in the special case that the variational approximation has the following form:

$$q_\phi(z|x) \propto h(z) \exp(\langle \psi(z), \text{NN}_\phi^{(i)}(x) \rangle) \tag{10}$$

where $\psi(z)$ are the features of z. If h is constant – which is true for a number of exponential family distributions such as the Bernoulli, Exponential, Pareto, Laplace, Gaussian, Gamma and the Wishart distributions – then:

$$q_\phi(z|x) \propto \exp(\langle \psi(z), \text{NN}_\phi^{(i)}(x) \rangle).$$

In turn,

$$\prod_{x \in \mathcal{Q}} q_\phi(z|x) \propto \exp(\langle \psi(z), \sum_{x \in \mathcal{Q}} \mathrm{NN}_\phi^{(i)}(x) \rangle).$$

This shows that the product of experts can be computed simply by summing the outputs of the neural network activations for such *deep-exponential* families with constant base measure.

C Bayesian Sets

The Bayesian Sets algorithm ranks the elements in $\mathcal{X} \setminus \mathcal{Q}$ according to the ratio of two probabilities:

$$score(x) = \frac{p(x|\mathcal{Q})}{p(x)} = \frac{E_{p(z|\mathcal{Q})}[p(x|z)]}{E_{\pi(z)}[p(x|z)]}$$

Instead of assuming the commonly used Beta-Binomial distribution we may assume that $p(x|z)$ is a product of independent Poisson distributions with Gamma conjugate priors. I.e. $p(x|z) = \prod_k \frac{z_k^{x_k}}{x_k}$. The conjugate prior on z is a product of Gamma distributions,

$$p(z|\alpha, \beta) = \prod_k \frac{{\beta_k}^{\alpha_k}}{\Gamma(\alpha_k)} {z_k}^{\alpha_k - 1} \exp(-\beta_k z_k)$$

. Let $f(x_k, \alpha_k, \beta_k) =$

$$\left(\frac{x_k + \alpha_k - 1}{x_k} \right) (1 - \frac{1}{1 + \beta_k})^{\alpha_k} (\frac{1}{1 + \beta_k})^{x_k}.$$

The Bayesian Sest score under these conditions is

$$score(x) = \prod_k \frac{f(x_k, \tilde{\alpha}_k, \tilde{\beta}_k)}{f(x_k, \alpha_k, \beta_k)}$$

Where $\tilde{\alpha}_k = \alpha_k + \sum_{x \in \mathcal{Q}} x_k$ and $\tilde{\beta}_k = \beta_k + \mathrm{Q}$. Note that if $\tilde{\alpha}_k = \alpha_k$ then $\frac{f(x_k, \tilde{\alpha}_k, \tilde{\beta}_k)}{f(x_k, \alpha_k, \beta_k)} = (\frac{1+\beta_k}{1+\beta_k+D})^{x_k}$ which means that features that occur in x that did not occur in $\mathcal{Q}$ are penalized based on the number of times the feature appeared. Therefore, the Gamma-Poisson distribution is a good approximation only when quantitative differences in the number of times a feature appears are important.

Finally we may assume that the components of x were sampled from conditionally independent gaussian distributions with unknown mean and precisions. I.e. $p(x|\mu, \tau) =$

$$\prod_k \sqrt{\frac{\tau}{2\pi}} \exp(-(x_k - \mu_k)^2 \tau_k)$$

and $p(\mu, \tau|\rho, \lambda, \alpha, \beta) =$

$$\prod_k \frac{{\beta_k}^{\alpha_k} \sqrt{\lambda_k}}{\Gamma(\alpha_k)\sqrt{2\pi}} {\tau_k}^{\alpha_k - \frac{1}{2}} \exp(-\beta_k \tau_k) \exp(-\frac{\lambda_k \tau_k (\mu_k - \rho_k)^2}{2}).$$

In the following formulaes we omit the susbscript k for convenience.

$$\begin{aligned}
\bar{x} &= \frac{1}{\mathrm{Q}} \sum_{x \in \mathcal{Q}} x \\
\tilde{\rho} &= \frac{\lambda \rho + \mathrm{Q} \bar{x}}{\lambda + \mathrm{Q}} \\
\tilde{\lambda} &= \lambda + \mathrm{Q} \\
\tilde{\alpha} &= \alpha + \mathrm{Q}/2 \\
\tilde{\beta} &= \beta + \frac{1}{2} \sum_{x \in \mathcal{Q}} (x - \bar{x})^2 + \frac{\mathrm{Q}\lambda}{\mathrm{Q} + \lambda} \frac{(\bar{x} - \tilde{\rho})^2}{2}
\end{aligned}$$

The Bayesian Sets score is the ratio of two t distribution values

$$score(x) = \prod_k \frac{t_{2\tilde{\alpha}_k}(x_k \mid \tilde{\rho}_k, \frac{\tilde{\beta}_k(\tilde{\lambda}_k+1)}{\tilde{\alpha}_k\tilde{\lambda}_k})}{t_{2\alpha_k}(x_k \mid \rho_k, \frac{\beta_k(\lambda_k+1)}{\alpha_k\lambda_k})}$$

Now the value of $t_\nu(x|a,b)$ where a is the location parameter and b is the scale parameter is:

$$t_\nu(x|a,b) = \frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{b\nu\pi}\Gamma(\frac{\nu}{2})}\left(1+\frac{(x-a)^2}{b\nu}\right)^{-\frac{\nu+1}{2}}$$

In order to use this distribution with count data, it is important to use some variance stabilizing transform, and then perform mean and variance normalization to preprocess all the count features. In this way we can set the priors $\tilde{\rho}_k$ to be 0 and λ_k can be set uniformly to some small number such as 2 and $alpha_k, \beta_k$ can be chosen to be $2, 1$ respectively.

C.1 Binarizing feature counts

BS binarizes the feature vector f_x as f'_x via thresholding:

$$f'_x[j] = \mathbb{I}[f_x[j] > \mu[j] + \lambda\sigma[j]]$$
$$\mu[j] = \frac{\sum_{x\in\mathcal{X}} f_x[j]}{\mathrm{X}}, \sigma^2[j] = \frac{\sum_{x\in\mathcal{X}}(f_x[j]-\mu[j])^2}{\mathrm{X}},$$

where $\lambda \in \mathbb{R}$ is a hyperparameter. BS's scoring function becomes

$$\underset{BS}{\text{score}}(\mathcal{Q},x) = \sum_{j=1}^{\mathrm{F}}\left(\log\frac{\tilde{\alpha}_\mathcal{Q}[j]\beta[j]}{\alpha[j]\tilde{\beta}_\mathcal{Q}[j]}\right)f'_x[j] \tag{11a}$$

$$\tilde{\alpha}_\mathcal{Q}[j] = \alpha[j] + \textstyle\sum_{x\in\mathcal{Q}} f'_x[j] \tag{11b}$$

$$\tilde{\beta}_\mathcal{Q}[j] = \beta[j] + \mathrm{Q} - \textstyle\sum_{x\in\mathcal{Q}} f'_x[j]. \tag{11c}$$

D Ranking methods

A standard function for computing the distance between distributions is the KL-divergence. Another possibility to compute the distance between distributions is to compute the symmetric version of the KL-divergence. Another standard method for computing the similarity between two probability distributions is to compute the probability product kernel (PPK) between two distributions [15]; i.e.

$$\langle q_\phi(z|\mathcal{Q}), q_\phi(z|x)\rangle = \int_z q_\phi(z|\mathcal{Q})q_\phi(z|x)dz$$

In the special case that $q_\phi(z|\mathcal{Q})$ and $q_\phi(z|x)$ have the special deep-gaussian form then the KL divergence as well as the inner product can be computed in closed form. KL Divergence between two distributions normal distributions p_1, p_2 with parameters (μ_1, Σ_1) and (μ_2, Σ_2) is:

$$KL(p_1||p_2) = \frac{1}{2}\left(\mathrm{tr}(\Sigma_2^{-1}\Sigma_1) + (\mu_1-\mu_2)^\top\Sigma_2^{-1}(\mu_1-\mu_2) - d + \log\frac{\det(\Sigma_2)}{\det(\Sigma_1)}\right).$$

and PPK is

$$\exp(\frac{-(\mu_1-\mu_2)^T(\Sigma_1+\Sigma_2)^{-1}(\mu_1-\mu_2)}{2} - \log\det((\Sigma_1+\Sigma_2)))$$

In the further special case that $\mu_2 = \mathbf{0}, \Sigma_2 = \mathbf{I}$ then the KL divergence simplifies to:

$$KL(p_1||p_2) = \frac{1}{2}\left(\mathrm{tr}(\Sigma_1) + \mu_1^T\mu_1 - d - \log\det(\Sigma_1)\right).$$

However, we propose here a simple way to compute the distance between two normal distributions. If μ_1, Σ_1 and μ_2, Σ_2 are the mean and variance of two normal distributions, p_1, p_2 then we use the following distance

$$d(p_1, p_2) = ||\mu_1\Sigma_1^{-1} - \mu_2\Sigma_2^{-1}||^2 = ||\xi_1 - \xi_2||^2$$

This metric can be implemented as a single matrix multiplication while KL divergence and PPK cannot. Intuitively this distance gives higher weightage to those dimensions where the variance of the either the distributions

is lower. In preliminary experiments we found this distance to be superior to KL divergence and PPL and we use this distance function in our experiments. We believe that the regularization from the gaussian prior that encourages the posterior distributions to be close to the origin make shift invariance unnecessary.

E Mechanical Turk HIT Interface and Extended Results

Table 5 shows the second and third place rankings of the systems and extends the results shown in Table 2.

Ents. In Query	Sents. Per Ent.	Group 1			Group 2		
		NVSE	BM25	BS	NVSE	SetEx	W2Vecf
3	1	36	28	16	20	21	39
	2-10	22	36	22	26	22	32
	11-100	24	26	30	23	22	34
5	1	28	37	15	20	47	13
	2-10	22	27	31	21	50	9
	11-100	20	27	32	17	29	34

Ents. In Query	Sents. Per Ent.	Group 1			Group 2		
		NVSE	BM25	BS	NVSE	SetEx	W2Vecf
3	1	17	14	49	9	45	26
	2-10	29	19	32	5	45	30
	11-100	21	31	28	12	48	20
5	1	14	18	48	2	14	64
	2-10	18	26	36	6	10	63
	11-100	36	20	24	11	40	29

Table 5: The number of times a system was ranked 2^{nd} (left subtable) and 3^{rd} (right subtable) over 80 queries.

Search #717

Query Entity	Description
"Iker Casillas"	"Iker Casillas Fernández (born 20 May 1981) is a Spanish football goalkeeper who plays for and captains both La Liga club Real Madrid and the Spanish national team. In 2008 he was the captain of the Spanish team that won their first European Championship in 44 years, the Spanish team that went on to win Spain's first World Cup (a tournament in which he won the Yashin Award) and the 2012 European Championship."
"Sergio Busquets"	"Sergio Busquets Burgos is a Spanish professional footballer who plays for FC Barcelona and the Spanish national team, as a defensive midfielder. He was a relatively obscure player when he arrived in FC Barcelona's first team in July 2008, but eventually made a name for himself in a relatively short period of time, reaching the Spanish national team in less than one year after making his professional club debut."
"Carles Puyol"	"Carles Puyol i Saforcada (born 13 April 1978) is a Spanish professional footballer who plays for FC Barcelona and the Spanish national team. Mainly a central defender he can also play on either flank, especially as a right back."

Results

System 1	System 2	System 3
"Xavi Hernandez" it was all change for barca at home to sevilla as well with players such as xavi hernandez sergio busquets pedro rodriguez jordi alba carles puyol	**"Jordi Alba"** should either of those two get injured or suspended before puyol is fit youngster marc bartra who clearly doesn t enjoy martinos	**"Barca"** real madrid coach jose mourinho insisted that his side have no chance of lifting the title as they are 15 points behind barca
"Barca" it was all change for barca at home to sevilla as well with players such as xavi hernandez sergio busquets pedro rodriguez jordi alba carles puyol	**"Xavi Hernandez"** meanwhile xavi hernandez will drag his vulnerable hamstring back into action and start in midfield alongside sergio busquets	**"Barack Obama"** the obama dogma of thought haters and haters of free speech demand illegal aliens to flood the usa to take americans and legal alien s jobs
"Leo Messi" messi had put barca 2 0 up by the break david villa put them ahead 10 minutes in the second half and jordi alba sealed the win by scoring on the break in the last minute of injury time	**"Iniesta"** meanwhile xavi hernandez will drag his vulnerable hamstring back into action and start in midfield alongside sergio busquets and andres iniesta	**"Leo Messi"** messi rattled the espanyol crossbar with a 30 yard free kick but there were no goals and barca will take their unbeaten record to malaga next week
"Iniesta" both barca and real madrid are likely to make wholesale changes for their weekend games with leo messi who looked a long way off full fitness on tuesday night given the chance to recover while others such as andres iniesta xavi hernandez and sergio busquets make way for players such as thiago alcantara david villa alex song and christian tello	**"Gerard Pique"** with central defender carles puyol still on his way back to fitness following a long term knee injury alba s absence is a real problem for coach tato martino given that adriano appears to currently be the first choice to cover for habitual central defensive pairing of gerard pique and javier mascherano	**"Donald Trump"** all of my blue collar friends are already voting for tru conservative ✓ 1 - best 2 3 - worst

Figure 3: Example of task shown to a crowd-source worker.