Autoencoding Pixies: Amortised Variational Inference with Graph Convolutions for Functional Distributional Semantics

Guy Emerson
Department of Computer Science and Technology
University of Cambridge
gete2@cam.ac.uk

Abstract

Functional Distributional Semantics provides a linguistically interpretable framework for distributional semantics, by representing the meaning of a word as a function (a binary classifier), instead of a vector. However, the large number of latent variables means that inference is computationally expensive, and training a model is therefore slow to converge. In this paper, I introduce the Pixie Autoencoder, which augments the generative model of Functional Distributional Semantics with a graph-convolutional neural network to perform amortised variational inference. This allows the model to be trained more effectively, achieving better results on two tasks (semantic similarity in context and semantic composition), and outperforming BERT, a large pre-trained language model.

1 Introduction

The aim of distributional semantics is to learn the meanings of words from a corpus (Harris, 1954; Firth, 1951, 1957). Many approaches learn a vector for each word, including count models and embedding models (for an overview, see: Erk, 2012; Clark, 2015), and some recent approaches learn a vector for each token in a particular context (for example: Peters et al., 2018; Devlin et al., 2019).

However, such vector representations do not make a clear distinction between words and the things they refer to. This means that such models are challenging to interpret semantically. In contrast, Functional Distributional Semantics (Emerson and Copestake, 2016) aims to provide a framework which can be interpreted in terms of model theory, a standard approach to formal semantics.

Furthermore, this framework supports first-order logic, where quantifying over logical variables is replaced by marginalising out random variables (Emerson and Copestake, 2017b; Emerson, 2020b). This connection to logic is a clear strength over vector-based models. Even the linguistically inspired tensor-based framework of Coecke et al. (2010) and Baroni et al. (2014) cannot model quantifiers, as shown by Grefenstette (2013).

However, the linguistic interpretability of Functional Distributional Semantics comes at a computational cost, with a high-dimensional latent variable for each token. Training a model by gradient descent requires performing Bayesian inference over these latent variables, which is intractable to calculate exactly. The main theoretical contribution of this paper is to present an amortised variational inference algorithm to infer these latent variables. This is done using a graph-convolutional network, as described in §3.

The main empirical contribution of this paper is to demonstrate that the resulting system, the Pixie Autoencoder, improves performance on two semantic tasks, as described in §4. I also present the first published results of applying a large language model (BERT) to these tasks, showing that results are sensitive to linguistic detail in how the model is applied. Despite being a smaller model trained on less data, the Pixie Autoencoder outperforms BERT on both tasks.

While the proposed inference network is designed for Functional Distributional Semantics, the proposed techniques should also be of wider interest. From a machine learning perspective, amortised variational inference with graph convolutions (§3.3) could be useful in other tasks where the input data is a graph, and the use of belief propagation to reduce variance (§3.4) could be useful for training other generative models. However, the most important contribution of this work is from a computational semantics perspective. This paper takes an important step towards truth-conditional distributional semantics, showing that truth-conditional functions can be efficiently learnt from a corpus.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 3982–3995
July 5 - 10, 2020.

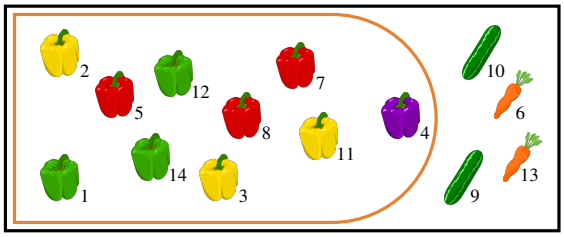

Figure 1: An example model structure with 14 individuals. Subscripts distinguish individuals with identical features, but are otherwise arbitrary. The *pepper* predicate is true of individuals inside the orange line, but the positions of individuals are otherwise arbitrary.

2 Functional Distributional Semantics

In this section, I summarise previous work on Functional Distributional Semantics. I begin in §2.1 by introducing model-theoretic semantics, which motivates the form of the machine learning model. I then explain in §2.2 how the meaning of a word is represented as a binary classifier, and finally present the probabilistic graphical model in §2.3.

2.1 Model-Theoretic Semantics

The basic idea of model-theoretic semantics is to define meaning in terms of *truth*, relative to *model structures*. A model structure can be understood as a model of the world. In the simplest case, it consists of a set of *individuals* (also called *entities*), as illustrated in Fig. 1. The meaning of a content word is called a *predicate*, and is formalised as a *truth-conditional function*, which maps individuals to *truth values* (either *truth* or *falsehood*).

Because of this precisely defined notion of truth, model theory naturally supports logic, and has become a prominent approach to formal semantics. For example, if we know the truth-conditional functions for *pepper* and *red*, we can use first-order logic to calculate the truth of sentences like *Some peppers are red*, for model structures like Fig. 1.

For detailed expositions, see: Cann (1993); Allan (2001); Kamp and Reyle (2013).

2.2 Semantic Functions

Functional Distributional Semantics (Emerson and Copestake, 2016; Emerson, 2018) embeds model-theoretic semantics into a machine learning model. An individual is represented by a feature vector, called a *pixie*.[1] For example, all three red pepper individuals in Fig. 1 would be represented by the same pixie, as they have the same features. A predicate is represented by a *semantic function*, which maps pixies to probabilities of truth. For example, the function for *pepper* should map the red pepper pixie to a probability close to 1. This can be seen in formal semantics as a truth-conditional function, and in a machine learning as a binary classifier.

This ties in with a view of concepts as abilities, as proposed in some schools of philosophy (for example: Dummett, 1976, 1978; Kenny, 2010; Sutton, 2015, 2017), and some schools of cognitive science (for example: Labov, 1973; McCloskey and Glucksberg, 1978; Murphy, 2002, pp. 1–3, 134–138; Zentall et al., 2002). In NLP, some authors have suggested representing concepts as classifiers, including Larsson (2013), working in the framework of Type Theory with Records (Cooper, 2005; Cooper et al., 2015). Similarly, Schlangen et al. (2016) and Zarrieß and Schlangen (2017a,b) train image classifiers using captioned images.

We can also view such a classifier as defining a region in the space, as argued for by Gärdenfors (2000, 2014). This idea is used for distributional semantics by Erk (2009a,b), for colour terms by McMahan and Stone (2015), and for knowledge base completion by Bouraoui et al. (2017).

For a broader survey motivating the use of classifiers to represent meaning, see: Emerson (2020a).

2.3 Probabilistic Graphical Model

To learn semantic functions in distributional semantics, Emerson and Copestake define a probabilistic graphical model that generates semantic dependency graphs, shown in Fig. 3. The basic idea is that an observed dependency graph is true of some unobserved situation comprising a number of individuals. Given a sembank (a corpus parsed into dependency graphs), the model can be trained unsupervised, to maximise the likelihood of generating the data. An example graph is shown in Fig. 2, which corresponds to sentences like *Every picture tells a story* or *The story was told by a picture* (note that only content words have nodes).

More precisely, given a *graph topology* (a dependency graph where the edges are labelled but the nodes are not), the model generates a predicate for each node. Rather than directly generating predicates, the model assumes that each predicate describes an unobserved individual.[2] The model

[1] Terminology introduced by Emerson and Copestake (2017a). This provides a useful shorthand for "feature representation of an individual".

[2] This assumes a neo-Davidsonian approach to event semantics (Davidson, 1967; Parsons, 1990), where verbal predicates are true of event individuals. It also assumes that a plural

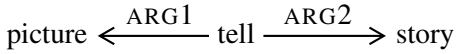

Figure 2: A dependency graph, which could be generated by Fig. 3. Such graphs are observed in training.

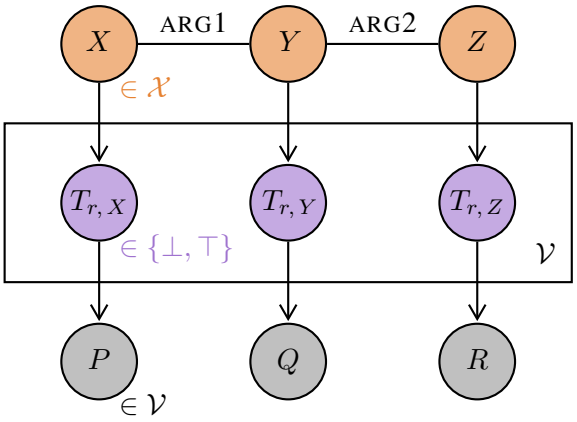

Figure 3: Probabilistic graphical model for Functional Distributional Semantics. Each node is a random variable. The plate (box in middle) denotes repeated nodes. **Top row:** individuals represented by jointly distributed pixie-valued random variables X, Y, Z, in a space $\mathcal{X}$. This is modelled by a Cardinality Restricted Boltzmann Machine (CaRBM), matching the graph topology. **Middle row:** for each individual, each predicate r in the vocabulary $\mathcal{V}$ is randomly true ($\top$) or false ($\bot$), according to the predicate's semantic function. Each function is modelled by a feedforward neural net. **Bottom row:** for each individual, we randomly generate one predicate, out of all predicates true of the individual. Only these nodes are observed.

first generates a pixie to represent each individual, then generates a truth value for each individual and each predicate in the vocabulary, and finally generates a single predicate for each individual. The pixies and truth values can be seen as a probabilistic model structure, which supports a probabilistic first-order logic (Emerson and Copestake, 2017b; Emerson, 2020b). This is an important advantage over other approaches to distributional semantics.

A pixie is defined to be a sparse binary-valued vector, with D *units* (dimensions), of which exactly C are *active* (take the value 1).[3] The joint distribution over pixies is defined by a Cardinality Restricted Boltzmann Machine (CaRBM) (Swersky et al., 2012), which controls how the active units of each pixie should co-occur with the active units of other pixies in the same dependency graph.

A CaRBM is an energy-based model, meaning that the probability of a situation is proportional to the exponential of the negative *energy* of the situation. This is shown in (1), where s denotes a *situation* comprising a set of pixies with semantic dependencies between them, and $E(s)$ denotes the energy. The energy is defined in (2),[4] where $x \xrightarrow{l} y$ denotes a dependency from pixie x to pixie y with label l. The CaRBM includes a weight matrix $w^{(l)}$ for each label l. The entry $w^{(l)}_{ij}$ controls how likely it is for units i and j to both be active, when linked by dependency l. Each graph topology has a corresponding CaRBM, but the weight matrices are shared across graph topologies. Normalising the distribution in (2) is intractable, as it requires summing over all possible s.

$$\mathbb{P}(s) \propto \exp\left(-E(s)\right) \tag{1}$$

$$\mathbb{P}(s) \propto \exp\left(\sum_{x \xrightarrow{l} y \text{ in } s} w^{(l)}_{ij} x_i y_j\right) \tag{2}$$

The semantic function $t^{(r)}$ for a predicate r is defined to be one-layer feedforward net, as shown in (3), where σ denotes the sigmoid function. Each predicate has a vector of weights $v^{(r)}$.

$$t^{(r)}(x) = \sigma\left(v^{(r)}_i x_i\right) \tag{3}$$

Lastly, the probability of generating a predicate r for a pixie x is given in (4). The more likely r is to be true, the more likely it is to be generated. Normalising requires summing over the vocabulary.

$$\mathbb{P}(r \mid x) \propto t^{(r)}(x) \tag{4}$$

In summary, the model has parameters $w^{(l)}$ (the world model), and $v^{(r)}$ (the lexical model). These are trained on a sembank using the gradients in (5), where g is a dependency graph. For $w^{(l)}$, only the first term is nonzero; for $v^{(r)}$, only the second term.

$$\frac{\partial}{\partial\theta}\log\mathbb{P}(g) = \left(\mathbb{E}_{s|g} - \mathbb{E}_s\right)\left[\frac{\partial}{\partial\theta}\left(-E(s)\right)\right] + \mathbb{E}_{s|g}\left[\frac{\partial}{\partial\theta}\log\mathbb{P}(g \mid s)\right] \tag{5}$$

noun corresponds to a plural individual, which would be compatible with Link (1983)'s approach to plural semantics.

[3] Although a pixie is a feature vector, the features are all latent in distributional semantics, in common with models like LDA (Blei et al., 2003) or Skip-gram (Mikolov et al., 2013).

[4] I follow the Einstein summation convention, where a repeated subscript is assumed to be summed over. For example, $x_i y_i$ is a dot product. Furthermore, I use uppercase for random variables, and lowercase for values. I abbreviate $\mathbb{P}(X = x)$ as $\mathbb{P}(x)$, and I abbreviate $\mathbb{P}(T_{r,X} = \top)$ as $\mathbb{P}(t_{r,X})$.

3 The Pixie Autoencoder

A practical challenge for Functional Distributional Semantics is training a model in the presence of high-dimensional latent variables. In this section, I present the Pixie Autoencoder, which augments the generative model with an encoder that predicts these latent variables.

For example, consider dependency graphs for *The child cut the cake* and *The gardener cut the grass*. These are true of rather different situations. Although the same verb is used in each, the pixie for *cut* should be different, because they describe events with different physical actions and different tools (slicing with a knife vs. driving a lawnmower). Training requires inferring posterior distributions for these pixies, but exact inference is intractable.

In §3.1 and §3.2, I describe previous work: amortised variational inference is useful to efficiently predict latent variables; graph convolutions are useful when the input is a graph. In §3.3, I present the encoder network, to predict latent pixies in Functional Distributional Semantics. It uses the tools introduced in §3.1 and §3.2, but modified to better suit the task. In §3.4, I explain how the encoder network can be used to train the generative model, since training requires the latent variables. Finally, I summarise the architecture in §3.5, and compare it to other autoencoders in §3.6.

3.1 Amortised Variational Inference

Calculating the gradients in (5) requires taking expectations over situations (both the marginal expectation $\mathbb{E}_s$, and the conditional expectation $\mathbb{E}_{s|g}$ given a graph). Exact inference would require summing over all possible situations, which is intractable for a high-dimensional space.

This is a general problem when working with probabilistic models. Given an intractable distribution $\mathbb{P}(x)$, a *variational inference* algorithm approximates this by a simpler distribution $\mathbb{Q}(x)$, parametrised by q, and then optimises the parameters so that $\mathbb{Q}$ is as close as possible to $\mathbb{P}$, where closeness is defined using KL-divergence (for a detailed introduction, see: Jordan et al., 1999).

However, variational inference algorithms typically require many update steps in order to optimise the approximating distribution $\mathbb{Q}$. An *amortised variational inference* algorithm makes a further approximation, by estimating the parameters q using an inference network (Kingma and Welling, 2014; Rezende et al., 2014; Titsias and Lázaro-Gredilla, 2014). The inference network might not predict the optimal parameters, but the calculation can be performed efficiently, rather than requiring many update steps. The network has its own parameters ϕ, which are optimised so that it makes good predictions for the variational parameters q.

3.2 Graph Convolutions

For graph-structured input data, a standard feedforward neural net is not suitable. In order to share parameters across similar graph topologies, an appropriate architecture is a *graph-convolutional* network (Duvenaud et al., 2015; Kearnes et al., 2016; Kipf and Welling, 2017; Gilmer et al., 2017). This produces a vector representation for each node in the graph, calculated through a number of layers. The vector for a node in layer k is calculated based only on the vectors in layer $k-1$ for that node and the nodes connected to it. The same weights are used for every node in the graph, allowing the network to be applied to different graph topologies.

For linguistic dependency graphs, the dependency labels carry important information. Marcheggiani and Titov (2017) propose using a different weight matrix for each label in each direction. This is shown in (6), where: $h^{(k,X)}$ denotes the vector representation of node X in layer k; $w^{(k,l)}$ denotes the weight matrix for dependency label l in layer k; f is a non-linear activation function; and the sums are over outgoing and incoming dependencies.[5] There is a separate weight matrix $w^{(k,l^{-1})}$ for a dependency in the opposite direction, and as well as a matrix $w^{(k,\text{self})}$ for updating a node based on itself. Bias terms are not shown.

$$h_i^{(k,X)} = f\Bigg(w_{ij}^{(k,\text{self})} h_j^{(k-1,X)} + \sum_{Y \xleftarrow{l} X} w_{ij}^{(k,l)} h_j^{(k-1,Y)} + \sum_{Y \xrightarrow{l} X} w_{ij}^{(k,l^{-1})} h_j^{(k-1,Y)} \Bigg) \tag{6}$$

3.3 Predicting Pixies

For Functional Distributional Semantics, Emerson and Copestake (2017a) propose a *mean-field* variational inference algorithm, where $\mathbb{Q}$ has an independent probability $q_i^{(X)}$ of each unit i being active, for each node X. Each probability is optimised based on the mean activation of all other units.

[5] For consistency with Fig. 3, I write X for a node (a random variable), rather than x (a pixie).

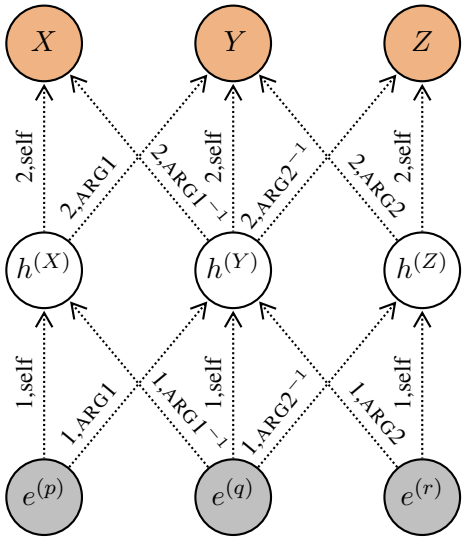

Figure 4: Graph-convolutional inference network for Fig. 3. The aim is to predict the posterior distribution over the pixie nodes X, Y, Z, given the observed predicates p, q, r. Each edge indicates the weight matrix used in the graph convolution, as defined in (6). In the bottom row, the input at each node is an embedding for the node's predicate. The intermediate representations h do not directly correspond to any random variables in Fig. 3. Conversely, the truth-valued random variables in Fig. 3 are not directly represented here.

This makes the simplifying assumption that the posterior distribution can be approximated as a single situation with some uncertainty in each dimension. For example, for a dependency graph for *The gardener cut the grass*, three mean vectors are inferred, for the gardener, the cutting event, and the grass. These vectors are "contextualised", because they are jointly inferred based on the whole graph.

I propose using a graph-convolutional network to amortise the inference of the variational mean-field vectors. In particular, I use the formulation in (6), with two layers. The first layer has a tanh activation, and the second layer has a sigmoid (to output probabilities). In addition, if the total activation in the second layer is above the total cardinality C, the activations are normalised to sum to C. The network architecture is illustrated in Fig. 4.

The network is trained to minimise the KL-divergence from $\mathbb{P}(s \mid g)$ (defined by the generative model) to $\mathbb{Q}(s)$ (defined by network's output). This is shown in (7), where $\mathbb{E}_{\mathbb{Q}(s)}$ denotes an expectation over s under the variational distribution.

$$D(\mathbb{Q}\|\mathbb{P}) = -\mathbb{E}_{\mathbb{Q}(s)}\left[\log\left(\frac{\mathbb{P}(s \mid g)}{\mathbb{Q}(s)}\right)\right] \tag{7}$$

To minimise the KL-divergence, we can differentiate with respect to the inference network parameters ϕ. This gives (8), where H denotes entropy.

$$\begin{aligned}\frac{\partial}{\partial\phi}D(\mathbb{Q}\|\mathbb{P}) = &-\frac{\partial}{\partial\phi}\mathbb{E}_{\mathbb{Q}(s)}\left[\log\mathbb{P}(s)\right] \\ &-\frac{\partial}{\partial\phi}\mathbb{E}_{\mathbb{Q}(s)}\left[\log\mathbb{P}(g \mid s)\right] \\ &-\frac{\partial}{\partial\phi}H(\mathbb{Q})\end{aligned} \tag{8}$$

The first term can be calculated exactly, because the log probability is proportional to the negative energy, which is a linear function of each pixie, and the normalisation constant is independent of s and $\mathbb{Q}$. This term therefore simplifies to the energy of the mean-field pixies, $\frac{\partial}{\partial\phi}E(\mathbb{E}[s])$.

The last term can be calculated exactly, because $\mathbb{Q}$ was chosen to be simple. Since each dimension is independent, it is $\sum_q q\log q + (1-q)\log(1-q)$, summing over the variational parameters.

The second term is more difficult, for two reasons. Firstly, calculating the probability of generating a predicate requires summing over all predicates, which is computationally expensive. We can instead sum over a random sample of predicates (along with the observed predicate). However, by ignoring most of the vocabulary, this will overestimate the probability of generating the correct predicate. I have mitigated this by upweighting this term, similarly to a β-VAE (Higgins et al., 2017).

The second problem is that the log probability of a predicate being true is not a linear function of the pixie. The first-order approximation would be to apply the semantic function to the mean-field pixie, as suggested by Emerson and Copestake (2017a). However, this is a poor approximation when the distribution over pixies has high variance. By approximating a sigmoid using a probit and assuming the input is approximately Gaussian, we can derive (9) (Murphy, 2012, §8.4.4.2). Intuitively, the higher the variance, the closer the expected value to $1/2$. For a Bernoulli distribution with probability q, scaled by a weight v, the variance is $v^2q(1–q)$.

$$\mathbb{E}[\sigma(x)] \approx \sigma\left(\frac{\mathbb{E}[x]}{\sqrt{1+\frac{\pi}{8}\mathrm{Var}[x]}}\right) \tag{9}$$

With the above approximations, we can calculate (4) efficiently. However, because the distribution over predicates in (4) only depends on *relative* probabilities of truth, the model might learn to keep them all close to 0, which would damage the logical interpretation of the model. To avoid this, I

have modified the second term of (5) and second term of (8), using not only the probability of *generating* a predicate for a pixie, $\mathbb{P}(r \mid x)$, but also the probability of the *truth* of a predicate, $\mathbb{P}(t_{r,X} \mid x)$. This technique of constraining latent variables to improve interpretability is similar to how Rei and Søgaard (2018) constrain attention weights.

Finally, as with other autoencoder models, there is a danger of learning an identity function that generalises poorly. Here, the problem is that the pixie distribution for a node might be predicted based purely on the observed predicate for that node, ignoring the wider context. To avoid this problem, we can use *dropout* on the input, a technique which has been effective for other NLP models (Iyyer et al., 2015; Bowman et al., 2016), and which is closely related to denoising autoencoders (Vincent et al., 2008). More precisely, we can keep the graph topology intact, but randomly mask out the predicates for some nodes. For a masked node X, I have initialised the encoder with an embedding as shown in (10), which depends on the node's dependencies (only on the label of each dependency, not on the predicate of the other node).

$$e^{(X)} = e^{(\text{drop})} + \sum_{Y \xleftarrow{l} X} e^{(\text{drop},l)} + \sum_{Y \xrightarrow{l} X} e^{(\text{drop},l^{-1})} \quad (10)$$

3.4 Approximating the Prior Expectation

The previous section explains the inference network and how it is trained. To train the generative model, the predictions of the inference network (without dropout) are used to approximate the conditional expectations $\mathbb{E}_{s|g}$ in (5). However, the prior expectation $\mathbb{E}_s$ cannot be calculated using the inference network. Intuitively, the prior distribution encodes a world model, and this cannot be summarised as a single mean-field situation.

Emerson and Copestake (2016) propose an MCMC algorithm using persistent particles, summing over samples to approximate the expectation. Many samples are required for a good approximation, which is computationally expensive. Taking a small number produces high variance gradients, which makes training less stable.

However, we can see in (5) that we don't need the prior expectation $\mathbb{E}_s$ on its own, but rather the difference $\mathbb{E}_{s|g} - \mathbb{E}_s$. So, to reduce the variance of gradients, we can try to explore the prior distribution only in the vicinity of the inference network's predictions. In particular, I propose taking the inference network's predictions and updating this mean-field distribution to bring it closer to the prior under the generative model. This can be done using belief propagation (for an introduction, see: Yedidia et al., 2003), as applied to CaRBMs by Swersky et al. (2012). For example, given the predicted mean-field vectors for a gardener cutting grass, we would modify these vectors to make the distribution more closely match what is plausible under the generative model (based on the world model, ignoring the observed predicates).

This can be seen as the bias-variance trade-off: the inference network introduces a bias, but reduces the variance, thereby making training more stable.

3.5 Summary

The Pixie Autoencoder is a combination of the generative model from Functional Distributional Semantics (generating dependency graphs from latent situations) and an inference network (inferring latent situations from dependency graphs), as illustrated in Figs. 3 and 4. They can be seen as an decoder and encoder, respectively.

It is trained on a sembank, with the generative model maximising the likelihood of the dependency graphs, and the inference network minimising KL-divergence with the generative model. To calculate gradients, the inference network is first applied to a dependency graph to infer the latent situation. The generative model gives the energy of the situation and the likelihood of the observed predicates (compared with random predicates). We also calculate the entropy of the situation, and apply belief propagation to get a situation closer to the prior. This gives us all terms in (5) and (8).

A strength of the Pixie Autoencoder is that it supports logical inference, following Emerson and Copestake (2017a). This is illustrated in Fig. 5. For example, for a gardener cutting grass or a child cutting a cake, we could ask whether the cutting event is also a slicing event or a mowing event.

3.6 Structural Prior

I have motivated the Pixie Autoencoder from the perspective of the generative model. However, we can also view it from the perspective of the encoder, comparing it with a Variational Autoencoder (VAE) which uses an RNN to generate text from a latent vector (Bowman et al., 2016). The VAE uses a Gaussian prior, but the Pixie Autoencoder has a structured prior defined by the world model.

Hoffman and Johnson (2016) find that VAEs struggle to fit a Gaussian prior. In contrast, the

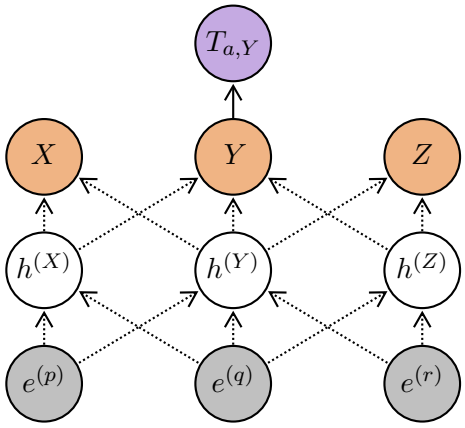

Figure 5: An example of logical inference, building on Fig. 4. Given an observed semantic dependency graph (here, with three nodes, like Fig. 2, with predicates p, q, r), we would like to know if some predicate is true of some latent individual (here, if a is true of Y). We can apply the inference network to infer distributions for the pixie nodes, and then apply a semantic function to a pixie node (here, the function for a applied to Y).

Pixie Autoencoder *learns* the prior, fitting the world model to the inference network's predictions. Since the world model makes structural assumptions, defining energy based only on semantic dependencies, we can see the world model as a "structural prior": the inference network is encouraged, via the first term in (8), to make predictions that can be modelled under these structural assumptions.

4 Experiments and Evaluation

I have evaluated on two datasets, chosen for two reasons. Firstly, they allow a direct comparison with previous results (Emerson and Copestake, 2017b). Secondly, they require fine-grained semantic understanding, which starts to use the expressiveness of a functional model.

More open-ended tasks such as lexical substitution and question answering would require combining my model with additional components such as a semantic parser and a coreference resolver. Robust parsers exist which are compatible with my model (for example: Buys and Blunsom, 2017; Chen et al., 2018), but this would be a non-trivial extension, particularly for incorporating robust coreference resolution, which would ideally be done hand-in-hand with semantic analysis. Incorporating fine-grained semantics into such tasks is an exciting research direction, but beyond the scope of the current paper.

When reporting results, significance tests follow Dror et al. (2018).

4.1 Training Details

I trained the model on WikiWoods (Flickinger et al., 2010; Solberg, 2012), which provides DMRS graphs (Copestake et al., 2005; Copestake, 2009) for 55m sentences (900m tokens) from the English Wikipedia (July 2008). It was parsed with the English Resource Grammar (ERG) (Flickinger, 2000, 2011) and PET parser (Callmeier, 2001; Toutanova et al., 2005), with parse ranking trained on WeScience (Ytrestøl et al., 2009). It is updated with each ERG release; I used the 1212 version. I preprocessed the data following Emerson and Copestake (2016), giving 31m graphs.

I implemented the model using DyNet (Neubig et al., 2017) and Pydmrs (Copestake et al., 2016).[6] I initialised the generative model following Emerson and Copestake (2017b) using sparse PPMI vectors (QasemiZadeh and Kallmeyer, 2016). I first trained the encoder on the initial generative model, then trained both together. I used L2 regularisation and the Adam optimiser (Kingma and Ba, 2015), with separate L2 weights and learning rates for the world model, lexical model, and encoder. I tuned hyperparameters on the RELPRON dev set (see §4.3), and averaged over 5 random seeds.

4.2 BERT Baseline

BERT (Devlin et al., 2019) is a large pre-trained language model with a Transformer architecture (Vaswani et al., 2017), trained on 3.3b tokens from the English Wikipedia and BookCorpus (Zhu et al., 2015). It produces high-quality contextualised embeddings, but its architecture is not motivated by linguistic theory. I used the version in the Transformers library (Wolf et al., 2019). To my knowledge, large language models have not previously been evaluated on these datasets.

4.3 RELPRON

The RELPRON dataset (Rimell et al., 2016) consists of *terms* (such as *telescope*), paired with up to 10 *properties* (such as *device that astronomer use*). The task is to find the correct properties for each term. There is large gap between the state of the art (around 50%) and the human ceiling (near 100%).

The dev set contains 65 terms and 518 properties; the test set, 73 terms and 569 properties. The dataset is too small to train on, but hyperparameters can be tuned on the dev set. The dev and test terms are disjoint, to avoid high scores from overtuning.

[6] https://gitlab.com/guyemerson/pixie

	Model	Dev	Test
Previous work	Vector addition (Rimell et al., 2016)	.496	.472
	Simplified Practical Lexical Function (Rimell et al., 2016)	.496	.497
	Vector addition (Czarnowska et al., 2019)	.485	.475
	Dependency vector addition (Czarnowska et al., 2019)	.497	.439
	Semantic functions (Emerson and Copestake, 2017b)	.20	.16
	Sem-func & vector ensemble (Emerson and Copestake, 2017b)	.53	.49
Baselines	Vector addition	.488	.474
	BERT (masked prediction)	.206	.186
	BERT (contextual prediction)	.093	.134
	BERT (masked prediction) & vector addition ensemble	.498	.479
Proposed approach	Pixie Autoencoder	.261	.189
	Pixie Autoencoder & vector addition ensemble	.532	.489

Table 1: Mean Average Precision (MAP) on RELPRON development and test sets.

Previous work has shown that vector addition performs well on this task (Rimell et al., 2016; Czarnowska et al., 2019). I have trained a Skip-gram model (Mikolov et al., 2013) using the Gensim library (Řehůřek and Sojka, 2010), tuning weighted addition on the dev set.

For the Pixie Autoencoder, we can view the task as logical inference, finding the probability of truth of a term given an observed property. This follows Fig. 5, applying the term a to either X or Z, according to whether the property has a subject or object relative clause.

BERT does not have a logical structure, so there are multiple ways we could apply it. I explored many options, to make it as competitive as possible. Following Petroni et al. (2019), we can rephrase each property as a cloze sentence (such as *a device that an astronomer uses is a [MASK]* .). However, RELPRON consists of pseudo-logical forms, which must be converted into plain text query strings. For each property, there are many possible cloze sentences, which yield different predictions. Choices include: grammatical number, articles, relative pronoun, passivisation, and position of the mask. I used the Pattern library (Smedt and Daelemans, 2012) to inflect words for number.

Results are given in Table 1. The best performing BERT method uses singular nouns with *a/an*, despite sometimes being ungrammatical. My most careful approach involves manually choosing articles (e.g. *a device*, *the sky*, *water*) and number (e.g. plural *people*) and trying three articles for the masked term (*a*, *an*, or no article, taking the highest probability from the three), but this actually lowers dev set performance to .192. Using plurals lowers performance to .089. Surprisingly, using BERT large (instead of BERT base) lowers performance to .165. As an alternative to cloze sentences, BERT can be used to predict the term from a contextualised embedding. This performs worse (see Table 1), but the best type of query string is similar.

The Pixie Autoencoder outperforms previous work using semantic functions, but is still outperformed by vector addition. Combining it with vector addition in a weighted ensemble lets us test whether they have learnt different kinds of information. The ensemble significantly outperforms vector addition on the test set ($p < 0.01$ for a permutation test), while the BERT ensemble does not ($p > 0.2$). However, it performs no better than the ensemble in previous work. This suggests that, while the encoder has enabled the model to learn more information, the additional information is already present in the vector space model.

RELPRON also includes a number of *confounders*, properties that are challenging due to lexical overlap. For example, an *activity that soil supports* is *farming*, not *soil*. There are 27 confounders in the test set, and my vector addition model places all of them in the top 4 ranks for the confounding term. In contrast, the Pixie Autoencoder and BERT do not fall for the confounders, with a mean rank of 171 and 266, respectively.

Nonetheless, vector addition remains hard to beat. As vector space models are known to be good at topical relatedness (e.g. learning that *astronomer* and *telescope* are related, without necessarily learning how they are related), a tentative conclusion is that relatedness is missing from the contextualised models (Pixie Autoencoder and BERT). Finding a principled way to integrate a notion of "topic" would be an interesting task for future work.

	Model	Separate	Averaged
Previous work	Vector addition (Milajevs et al., 2014)	-	.348
	Categorical, copy object (Milajevs et al., 2014)	-	.456
	Categorical, regression (Polajnar et al., 2015)	.33	-
	Categorical, low-rank decomposition (Fried et al., 2015)	.34	-
	Tensor factorisation (Van de Cruys et al., 2013)	.37	-
	Neural categorical (Hashimoto et al., 2014)	.41	.50
	Semantic functions (Emerson and Copestake, 2017b)	.25	-
	Sem-func & vector ensemble (Emerson and Copestake, 2017b)	.32	-
Baselines	BERT (contextual similarity)	.337	.446
	BERT (contextual prediction)	.233	.317
Proposed approach	Pixie Autoencoder (logical inference in both directions)	.306	.374
	Pixie Autoencoder (logical inference in one direction)	.406	.504

Table 2: Spearman rank correlation on the GS2011 dataset, using separate or averaged annotator scores.

4.4 GS2011

The GS2011 dataset evaluates similarity in context (Grefenstette and Sadrzadeh, 2011). It comprises pairs of verbs combined with the same subject and object (for example, *map show location* and *map express location*), annotated with similarity judgements. There are 199 distinct pairs, and 2500 judgements (from multiple annotators).

Care must be taken when considering previous work, for two reasons. Firstly, there is no development set. Tuning hyperparameters directly on this dataset will lead to artificially high scores, so previous work cannot always be taken at face value. For example, Hashimoto et al. (2014) report results for 10 settings. I nonetheless show the best result in Table 2. My model is tuned on RELPRON (§4.3).

Secondly, there are two ways to calculate correlation with human judgements: averaging for each distinct pair, or keeping each judgement separate. Both methods have been used in previous work, and only Hashimoto et al. (2014) report both.

For the Pixie Autoencoder, we can view the task as logical inference, following Fig. 5. However, Van de Cruys et al. (2013) point out that the second verb in each pair is often nonsensical when combined with the two arguments (e.g. *system visit criterion*), and so they argue that only the first verb should be contextualised, and then compared with the second verb. This suggests we should apply logical inference only in one direction: we should find the probability of truth of the second verb, given the first verb and its arguments. As shown in Table 2, this gives better results than applying logical inference in both directions and averaging the probabilities. Logical inference in both directions allows a direct comparison with Emerson and Copestake (2017b), showing the Pixie Autoencoder performs better. Logical inference in one direction yields state-of-the-art results on par with the best results of Hashimoto et al. (2014).

There are multiple ways to apply BERT, as in §4.3. One option is to calculate cosine similarity of contextualised embeddings (averaging if tokenised into word-parts). However, each subject-verb-object triple must be converted to plain text. Without a dev set, it is reassuring that conclusions from RELPRON carry over: it is best to use singular nouns with *a/an* (even if ungrammatical) and it is best to use BERT base. Manually choosing articles and number lowers performance to .320 (separate), plural nouns to .175, and BERT large to .226. Instead of using cosine similarity, we can predict the other verb from the contextualised embedding, but this performs worse. The Pixie Autoencoder outperforms BERT, significantly for separate scores ($p < 0.01$ for a bootstrap test), but only suggestively for averaged scores ($p = 0.18$).

5 Conclusion

I have presented the Pixie Autoencoder, a novel encoder architecture and training algorithm for Functional Distributional Semantics, improving on previous results in this framework. For GS2011, the Pixie Autoencoder achieves state-of-the-art results. For RELPRON, it learns information not captured by a vector space model. For both datasets, it outperforms BERT, despite being a shallower model with fewer parameters, trained on less data. This points to the usefulness of building semantic structure into the model. It is also easy to apply to these datasets (with no need to tune query strings), as it has a clear logical interpretation.

References

Keith Allan. 2001. *Natural language semantics*. Blackwell.

Marco Baroni, Raffaela Bernardi, and Roberto Zamparelli. 2014. Frege in space: A program of compositional distributional semantics. *Linguistic Issues in Language Technology (LiLT)*, 9.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022.

Zied Bouraoui, Shoaib Jameel, and Steven Schockaert. 2017. Inductive reasoning about ontologies using conceptual spaces. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. Association for the Advancement of Artificial Intelligence.

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL)*, pages 10–21.

Jan Buys and Phil Blunsom. 2017. Robust incremental neural semantic graph parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL), Long Papers*, pages 1215–1226.

Ulrich Callmeier. 2001. Efficient parsing with large-scale unification grammars. Master's thesis, Saarland University.

Ronnie Cann. 1993. *Formal semantics: an introduction*. Cambridge University Press.

Yufei Chen, Weiwei Sun, and Xiaojun Wan. 2018. Accurate SHRG-based semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), Long Papers*, pages 408–418.

Stephen Clark. 2015. Vector space models of lexical meaning. In Shalom Lappin and Chris Fox, editors, *The Handbook of Contemporary Semantic Theory*, 2nd edition, chapter 16, pages 493–522. Wiley.

Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. 2010. Mathematical foundations for a compositional distributional model of meaning. *Linguistic Analysis*, 36, A Festschrift for Joachim Lambek:345–384.

Robin Cooper. 2005. Austinian truth, attitudes and type theory. *Research on Language and Computation*, 3(2-3):333–362.

Robin Cooper, Simon Dobnik, Staffan Larsson, and Shalom Lappin. 2015. Probabilistic type theory and natural language semantics. *Linguistic Issues in Language Technology (LiLT)*, 10.

Ann Copestake. 2009. Slacker semantics: Why superficiality, dependency and avoidance of commitment can be the right way to go. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 1–9.

Ann Copestake, Guy Emerson, Michael Wayne Goodman, Matic Horvat, Alexander Kuhnle, and Ewa Muszyńska. 2016. Resources for building applications with Dependency Minimal Recursion Semantics. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC)*, pages 1240–1247. European Language Resources Association (ELRA).

Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan A. Sag. 2005. Minimal Recursion Semantics: An introduction. *Research on Language and Computation*, 3(2-3):281–332.

Tim Van de Cruys, Thierry Poibeau, and Anna Korhonen. 2013. A tensor-based factorization model of semantic compositionality. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies*, pages 1142–1151.

Paula Czarnowska, Guy Emerson, and Ann Copestake. 2019. Words are vectors, dependencies are matrices: Learning word embeddings from dependency graphs. In *Proceedings of the 13th International Conference on Computational Semantics (IWCS), Long Papers*, pages 91–102. Association for Computational Linguistics.

Donald Davidson. 1967. The logical form of action sentences. In Nicholas Rescher, editor, *The Logic of Decision and Action*, chapter 3, pages 81–95. University of Pittsburgh Press. Reprinted in: Davidson (1980/2001), *Essays on Actions and Events*, Oxford University Press.

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

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), Volume 1 (Long Papers)*, pages 1383–1392.

Michael Dummett. 1976. What is a theory of meaning? (II). In Gareth Evans and John McDowell, editors, *Truth and Meaning: Essays in Semantics*, chapter 4, pages 67–137. Clarendon Press (Oxford). Reprinted in: Dummett (1993), *Seas of Language*, chapter 2, pages 34–93.

Michael Dummett. 1978. What do I know when I know a language? Presented at the Centenary Celebrations of Stockholm University. Reprinted in: Dummett (1993), *Seas of Language*, chapter 3, pages 94–105.

David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2224–2232.

Guy Emerson. 2018. *Functional Distributional Semantics: Learning Linguistically Informed Representations from a Precisely Annotated Corpus*. Ph.D. thesis, University of Cambridge.

Guy Emerson. 2020a. What are the goals of distributional semantics? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Guy Emerson. 2020b. Linguists who use probabilistic models love them: Quantification in Functional Distributional Semantics. In *Proceedings of Probability and Meaning (PaM2020)*. Association for Computational Linguistics.

Guy Emerson and Ann Copestake. 2016. Functional Distributional Semantics. In *Proceedings of the 1st Workshop on Representation Learning for NLP (RepL4NLP)*, pages 40–52. Association for Computational Linguistics.

Guy Emerson and Ann Copestake. 2017a. Variational inference for logical inference. In *Proceedings of the Conference on Logic and Machine Learning in Natural Language (LaML)*, pages 53–62. Centre for Linguistic Theory and Studies in Probability (CLASP).

Guy Emerson and Ann Copestake. 2017b. Semantic composition via probabilistic model theory. In *Proceedings of the 12th International Conference on Computational Semantics (IWCS)*, pages 62–77. Association for Computational Linguistics.

Katrin Erk. 2009a. Supporting inferences in semantic space: representing words as regions. In *Proceedings of the 8th International Conference on Computational Semantics (IWCS)*, pages 104–115. Association for Computational Linguistics.

Katrin Erk. 2009b. Representing words as regions in vector space. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL)*, pages 57–65. Association for Computational Linguistics.

Katrin Erk. 2012. Vector space models of word meaning and phrase meaning: A survey. *Language and Linguistics Compass*, 6(10):635–653.

John Rupert Firth. 1951. Modes of meaning. *Essays and Studies of the English Association*, 4:118–149. Reprinted in: Firth (1957), *Papers in Linguistics*, chapter 15, pages 190–215.

John Rupert Firth. 1957. A synopsis of linguistic theory 1930–1955. In John Rupert Firth, editor, *Studies in Linguistic Analysis*, Special volume of the Philological Society, chapter 1, pages 1–32. Blackwell.

Dan Flickinger. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering*, 6(1):15–28.

Dan Flickinger. 2011. Accuracy vs. robustness in grammar engineering. In Emily M. Bender and Jennifer E. Arnold, editors, *Language from a Cognitive Perspective: Grammar, Usage, and Processing*, chapter 3, pages 31–50. Center for the Study of Language and Information (CSLI) Publications.

Dan Flickinger, Stephan Oepen, and Gisle Ytrestøl. 2010. WikiWoods: Syntacto-semantic annotation for English Wikipedia. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC)*, pages 1665–1671. European Language Resources Association (ELRA).

Daniel Fried, Tamara Polajnar, and Stephen Clark. 2015. Low-rank tensors for verbs in compositional distributional semantics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL), Short Papers*, pages 731–736.

Peter Gärdenfors. 2000. *Conceptual spaces: The geometry of thought*. Massachusetts Institute of Technology (MIT) Press.

Peter Gärdenfors. 2014. *Geometry of meaning: Semantics based on conceptual spaces*. Massachusetts Institute of Technology (MIT) Press.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1263–1272.

Edward Grefenstette. 2013. Towards a formal distributional semantics: Simulating logical calculi with tensors. In *Proceedings of the 2nd Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 1–10. Association for Computational Linguistics.

Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the 16th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1394–1404. Association for Computational Linguistics.

Zellig Sabbetai Harris. 1954. Distributional structure. *Word*, 10:146–162. Reprinted in: Harris (1970), *Papers in Structural and Transformational Linguistics*, chapter 36, pages 775–794; Harris (1981), *Papers on Syntax*, chapter 1, pages 3-22.

Kazuma Hashimoto, Pontus Stenetorp, Makoto Miwa, and Yoshimasa Tsuruoka. 2014. Jointly learning word representations and composition functions using predicate-argument structures. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1544–1555.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*.

Matthew D Hoffman and Matthew J Johnson. 2016. ELBO surgery: yet another way to carve up the variational evidence lower bound. In *Proceedings of the NIPS 2016 Workshop on Advances in Approximate Bayesian Inference*.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL) and the 7th International Joint Conference on Natural Language Processing (IJCNLP), Volume 1 (Long Papers)*, volume 1, pages 1681–1691.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K Saul. 1999. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233.

Hans Kamp and Uwe Reyle. 2013. *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*, volume 42 of *Studies in Linguistics and Philosophy*. Springer.

Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. 2016. Molecular graph convolutions: moving beyond fingerprints. *Journal of Computer-Aided Molecular Design*, 30(8):595–608.

Anthony Kenny. 2010. Concepts, brains, and behaviour. *Grazer Philosophische Studien*, 81(1):105–113.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*.

William Labov. 1973. The boundaries of words and their meanings. In Charles-James Bailey and Roger W. Shuy, editors, *New Ways of Analyzing Variation in English*, chapter 24, pages 340–371. Georgetown University Press. Reprinted in: Ralph W. Fasold, editor (1983), *Variation in the Form and Use of Language: A Sociolinguistics Reader*, chapter 3, pages 29–62, Georgetown University Press.

Staffan Larsson. 2013. Formal semantics for perceptual classification. *Journal of Logic and Computation*, 25(2):335–369.

Godehard Link. 1983. The logical analysis of plurals and mass terms: A lattice-theoretical approach. In Rainer Bäuerle, Christoph Schwarze, and Arnim von Stechow, editors, *Meaning, Use and the Interpretation of Language*, chapter 18, pages 303–323. Walter de Gruyter. Reprinted in: Paul Portner and Barbara H. Partee, editors (2002), *Formal semantics: The essential readings*, chapter 4, pages 127–146.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1506–1515.

Michael E. McCloskey and Sam Glucksberg. 1978. Natural categories: Well defined or fuzzy sets? *Memory & Cognition*, 6(4):462–472.

Brian McMahan and Matthew Stone. 2015. A Bayesian model of grounded color semantics. *Transactions of the Association for Computational Linguistics (TACL)*, 3:103–115.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the 1st International Conference on Learning Representations (ICLR), Workshop Track*.

Dmitrijs Milajevs, Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Matthew Purver. 2014. Evaluating neural word representations in tensor-based compositional settings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 708–719.

Gregory Murphy. 2002. *The big book of concepts*. Massachusetts Institute of Technology (MIT) Press.

Kevin P. Murphy. 2012. *Machine Learning: A Probabilistic Perspective*. Massachusetts Institute of Technology (MIT) Press.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui,

Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. Dynet: The dynamic neural network toolkit. Unpublished manuscript, arXiv preprint 1701.03980.

Terence Parsons. 1990. *Events in the Semantics of English: A Study in Subatomic Semantics*. Current Studies in Linguistics. Massachusetts Institute of Technology (MIT) Press.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

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

Tamara Polajnar, Laura Rimell, and Stephen Clark. 2015. An exploration of discourse-based sentence spaces for compositional distributional semantics. In *Proceedings of the EMNLP Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics (LSDSem)*, pages 1–11. Association for Computational Linguistics.

Behrang QasemiZadeh and Laura Kallmeyer. 2016. Random positive-only projections: PPMI-enabled incremental semantic space construction. In *Proceedings of the 5th Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 189–198. Association for Computational Linguistics.

Radim Řehůřek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 46–50. European Language Resources Association (ELRA).

Marek Rei and Anders Søgaard. 2018. Zero-shot sequence labeling: Transferring knowledge from sentences to tokens. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies, Volume 1 (Long Papers)*, pages 293–302.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 1278–1286.

Laura Rimell, Jean Maillard, Tamara Polajnar, and Stephen Clark. 2016. RELPRON: A relative clause evaluation dataset for compositional distributional semantics. *Computational Linguistics*, 42(4):661–701.

David Schlangen, Sina Zarrieß, and Casey Kennington. 2016. Resolving references to objects in photographs using the words-as-classifiers model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), Long Papers*, pages 1213–1223.

Tom De Smedt and Walter Daelemans. 2012. Pattern for Python. *Journal of Machine Learning Research (JMLR)*, 13:2063–2067.

Lars Jørgen Solberg. 2012. A corpus builder for Wikipedia. Master's thesis, University of Oslo.

Peter R. Sutton. 2015. Towards a probabilistic semantics for vague adjectives. In Henk Zeevat and Hans-Christian Schmitz, editors, *Bayesian Natural Language Semantics and Pragmatics*, chapter 10, pages 221–246. Springer.

Peter R. Sutton. 2017. Probabilistic approaches to vagueness and semantic competency. *Erkenntnis*.

Kevin Swersky, Ilya Sutskever, Daniel Tarlow, Richard S. Zemel, Ruslan R. Salakhutdinov, and Ryan P. Adams. 2012. Cardinality Restricted Boltzmann Machines. In *Advances in Neural Information Processing Systems 25 (NIPS)*, pages 3293–3301.

Michalis Titsias and Miguel Lázaro-Gredilla. 2014. Doubly stochastic variational Bayes for non-conjugate inference. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 1971–1979.

Kristina Toutanova, Christopher D. Manning, Dan Flickinger, and Stephan Oepen. 2005. Stochastic HPSG parse selection using the Redwoods corpus. *Research on Language and Computation*, 3(1):83–105.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine learning (ICML)*, pages 1096–1103. Association for Computing Machinery (ACM).

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace

Transformers: State-of-the-art natural language processing. Unpublished manuscript, arXiv preprint 1910.03771.

Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. 2003. Understanding Belief Propagation and its generalizations. In Gerhard Lakemeyer and Bernhard Nebel, editors, *Exploring Artificial Intelligence in the New Millennium*, chapter 8, pages 239–269. Morgan Kaufmann Publishers.

Gisle Ytrestøl, Dan Flickinger, and Stephan Oepen. 2009. Extracting and annotating Wikipedia subdomains: Towards a new eScience community resource. In *Proceedings of the 7th International Workshop on Treebanks and Linguistic Theories*, pages 185–197.

Sina Zarrieß and David Schlangen. 2017a. Is this a child, a girl or a car? Exploring the contribution of distributional similarity to learning referential word meanings. In *Proceedings of the 15th Annual Conference of the European Chapter of the Association for Computational Linguistics (EACL), Short Papers*, pages 86–91.

Sina Zarrieß and David Schlangen. 2017b. Obtaining referential word meanings from visual and distributional information: Experiments on object naming. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL), Long Papers*, pages 243–254.

Thomas R. Zentall, Mark Galizio, and Thomas S. Critchfield. 2002. Categorization, concept learning, and behavior analysis: An introduction. *Journal of the Experimental Analysis of Behavior*, 78(3):237–248.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, pages 19–27.