Open Knowledge Graphs Canonicalization using Variational Autoencoders

Sarthak Dash, Gaetano Rossiello, Sugato Bagchi,
Nandana Mihindukulasooriya, Alfio Gliozzo
IBM Research AI, Thomas J. Watson Research Center
Yorktown Heights, NY

Abstract

Noun phrases and Relation phrases in open knowledge graphs are not canonicalized, leading to an explosion of redundant and ambiguous subject-relation-object triples. Existing approaches to solve this problem take a two-step approach. First, they generate embedding representations for both noun and relation phrases, then a clustering algorithm is used to group them using the embeddings as features. In this work, we propose Canonicalizing Using Variational Autoencoders (CUVA)[1], a joint model to learn both embeddings and cluster assignments in an end-to-end approach, which leads to a better vector representation for the noun and relation phrases. Our evaluation over multiple benchmarks shows that CUVA outperforms the existing state-of-the-art approaches. Moreover, we introduce CANONICNELL, a novel dataset to evaluate entity canonicalization systems.

1 Introduction

Open Information Extraction (OpenIE) methods (Fader et al., 2011a; Stanovsky et al., 2018) can be used to extract triples in the form *(noun phrase, relation phrase, noun phrase)* from given text corpora in an unsupervised way without requiring a pre-defined ontology schema. This makes them suitable to build large Open Knowledge Graphs (OpenKGs) from huge collections of unstructured text documents, thereby making the usage of OpenIE methods highly adaptable to new domains.

Although OpenIE methods are highly adaptable, one major shortcoming of OpenKGs is that Noun Phrases (NPs) and Relation Phrases (RPs) are not *canonicalized*. This means that two NPs (or RPs) having different surface forms, but referring to the same entity (or relation) in a canonical KB, are treated differently. Consider the following triples as an example: (NBC-TV, has headquarters in, NYC), (NBC Television, is in, New York City) and (NBC-TV, has main office in, NYC). Looking at the previous example, both OpenIE methods and associated Open KGs would not have any knowledge that *NYC* and *New York City* refer to the same entity, or *has headquarters in* and *has main office in* are similar relations.

Moreover, while it is true that similar relations will have same argument types (see the previous example), the converse need not hold true. For example, given the following two triples (X, is born in, Y) and (X, has died in, Y) in an Open KG, where X is of type *Person* and Y is of type *Location*, does not imply *is born in* and *has died in* are similar relations.

Thus, the task of *canonicalizing* NPs and RPs within an Open KG is significant. Otherwise, Open KGs will have an explosion of redundant facts, which is highly undesirable, for the following reasons. Firstly, redundant facts use a higher memory footprint. Secondly, querying an Open KG is likely to yield sub-optimal results, for e.g. it will not return all facts associated with *NYC* when using *New York City* as the query. Finally, allowing downstream applications such as Link Prediction (Bordes et al., 2013) to know that *NYC* and *New York City* refers to the same entity, will improve their performance while operating on large Open KGs. Hence, it is imperative to *canonicalize* NPs and RPs within an Open KG.

In this paper, we introduce Canonicalizing Using Variational Autoencoders (CUVA), a neural network architecture that learns unique embeddings for NPs and RPs as well as cluster assignments in a joint fashion. CUVA combines *a*) The Variational Deep Embedding (VaDE) framework (Jiang et al., 2017a), a generative approach to Clustering, and *b*) A KG Embedding Model that aims to utilize the structural knowledge present within the Open KG. In addition, CUVA uses additional contextual

[1] https://github.com/IBM/Open-KG-canonicalization

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 10379–10394
November 7–11, 2021. ©2021 Association for Computational Linguistics

information obtained from the documents used to build the Open KG.

The input to CUVA is *a*) An Open KG expressed as a list of triples and *b*) Contextual Information obtained from the documents. The output is a set of NP and RP clusters grouping all items together that refer to the same entity (or relation).

In summary, we make the following contributions,

- We introduce CUVA, a novel neural architecture for the CANONICALIZATION task, based on joint learning of mention representations and cluster assignments for entity and relation clusters using variational autoencoders.
- We demonstrate empirically that CUVA improves state of the art (SOTA) on the Entity CANONICALIZATION task, across four academic benchmarks.

2 Related Work

Extracting triples from sentences is the first step to build Open KGs. The OpenIE technique has been originally introduced in (Banko et al., 2007). Thereafter, several approaches have been proposed to improve the quality of the extracted triples. Rule-based approaches, such as REVERB (Fader et al., 2011a) and PREDPATT (White et al., 2016), use patterns on top of syntactic features to extract relation phrases and their arguments from text. Learning-based methods, such as OLLIE (Mausam et al., 2012) and RNNOIE (Stanovsky et al., 2018), train a self-supervised system using bootstrapping techniques. Clause-based approaches (Angeli et al., 2015) navigate through the dependency trees to split the sentences into simpler and independent segments.

There have been several previous works to group NPs and RPs into coherent clusters. A traditional approach to canonicalize NPs is to map them to an existing KB such as Wikidata, also referred to as the Entity Linking (EL) task (Lin et al., 2012; Ceccarelli et al., 2014). A major problem with these EL approaches is that many NPs may refer to entities that are not present in the KB, in which case they are not clustered.

The RESOLVER system (Yates and Etzioni, 2009) uses string similarity features to cluster phrases in TextRunner (Banko et al., 2007) triples. (Galárraga et al., 2014a) uses manually defined features for NP canonicalization, and subsequently performs relation phrase clustering by using AMIE algorithm (Galárraga et al., 2013). (Wu et al., 2018) propose a modification to the previous approach by using pruning and bounding techniques. Concept Resolver (Krishnamurthy and Mitchell, 2011), which makes “one sense per category” assumption, is used for clustering NP mentions in NELL. This approach requires additional information in the form of a schema of relation types. KB-Unify (Delli Bovi et al., 2015) addresses the problem of unifying multiple canonical and open KGs into one KG, but requires additional sense inventory, which may not be available.

The CESI architecture (Vashishth et al., 2018a) models the CANONICALIZATION task in a two-step pipeline approach, i.e., in the *first* step, it uses a HolE algorithm (Nickel et al., 2016) to learn embeddings for NPs (and RPs), and then in an *independent* second step, it “plugs" these learned embeddings into a Hierarchical Agglomerative Clustering (HAC) algorithm to generate clusters. Currently, CESI is the state of the art on this task. Unlike CESI, our proposed model CUVA learns the embedding representations and the cluster assignments of both NPs and RPs in an end-to-end manner, using a single model.

3 Open KGs Canonicalization Using VAE

Formally, the CANONICALIZATION task is defined as follows: given a list of triples $\mathcal{T} = (h, r, t)$ from an OpenIE system $\mathcal{O}$ on a document collection $\mathcal{C}$, where h, t are Noun Phrases (NPs) and r is a Relation Phrase (RP), the objective is to cluster NPs (and RPs), so that items referring to the same entity (or relation) are in the same cluster.

We assume that each cluster corresponds to either a *latent* entity or a *latent* relation; the label of such a *latent* entity/relation is unknown to the learner.

CUVA uses two variational autoencoders i.e. E-VAE and R-VAE, one each for *entities* and *relations*. Both E-VAE and R-VAE use a mixture of Gaussians for modeling *latent* entities and relations. Also, we use a Knowledge Graph Embedding (KGE) module to encode the structural information present within the Open KG. CUVA works as follows:

1. A *latent* entity (or relation) as defined above, is modeled via a Gaussian distribution. The sampled items from the Gaussian distribution

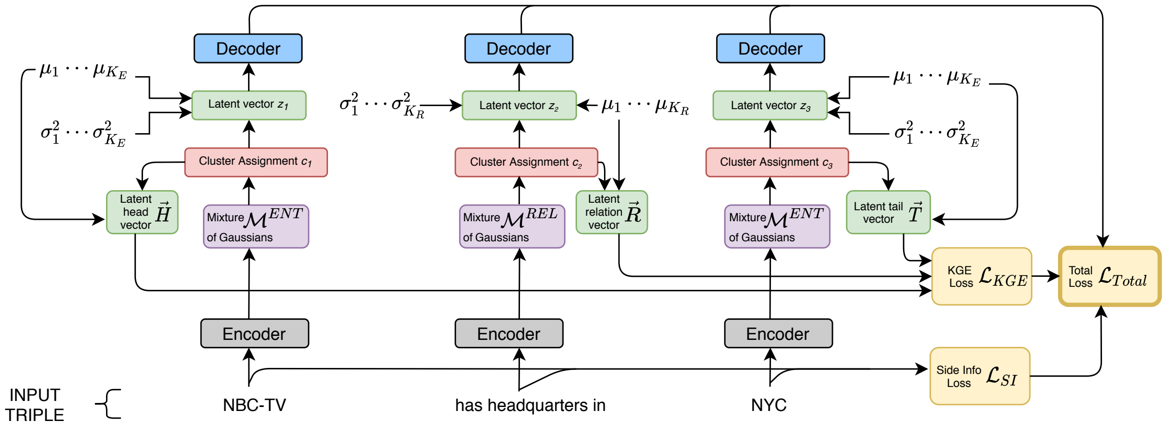
Decoder
Decoder
Decoder
$\mu_1 \cdots \mu_{K_E}$
$\sigma_1^2 \cdots \sigma_{K_E}^2$
Latent vector z_1
$\sigma_1^2 \cdots \sigma_{K_R}^2$
Latent vector z_2
$\mu_1 \cdots \mu_{K_R}$
Latent vector z_3
$\mu_1 \cdots \mu_{K_E}$
$\sigma_1^2 \cdots \sigma_{K_E}^2$
Cluster Assignment c_1
Cluster Assignment c_2
Cluster Assignment c_3
Latent head vector $\vec{H}$
Mixture of Gaussians $\mathcal{M}^{ENT}$
Mixture of Gaussians $\mathcal{M}^{REL}$
Latent relation vector $\vec{R}$
Mixture of Gaussians $\mathcal{M}^{ENT}$
Latent tail vector $\vec{T}$
KGE Loss $\mathcal{L}_{KGE}$
Total Loss $\mathcal{L}_{Total}$
Encoder
Encoder
Encoder
Side Info Loss $\mathcal{L}_{SI}$
INPUT TRIPLE
NBC-TV
has headquarters in
NYC

Figure 1: The core structure of CUVA. The *left* and *right* vertical structures correspond to the Entity variational autoencoder (E-VAE) for the head and tail Noun Phrases, whereas the *middle* vertical structure corresponds to the Relation variational autoencoder (R-VAE) for the Relation Phrase. The KGE module connects both the E-VAE and the R-VAE as shown above.

correspond to the observed NPs (and RPs) within $\mathcal{T}$.

2. NPs (h, t) and RPs (r) are modeled using larger embedding dimensions compared to the Gaussian distributions, to account for variations in the observed surface forms.

3. We use Gaussian parameters to refer to the entity (relation) as opposed to the NP (or RP).

4. Since the items are clustered together, we assume that different NPs, e.g. *New York City* and *NYC* (or RPs) belonging to the same Gaussian distribution (i.e. cluster) have similar *attributes*.

Fig. 1 illustrates an instantiation for CUVA. A description of each of the components of CUVA follows below.

3.1 Variational Autoencoder

Based on the above modeling assumptions, we use Variational Deep Embedding (VaDE) (Jiang et al., 2017a) generative model for clustering. This generative clustering model implements a Mixture of Gaussians within the latent space of a variational autoencoder (VAE). We believe such a model is better suited to cluster mentions because its soft-clustering ability can account for different senses (polysemy) of a given entity mention. Such behavior is preferable to hard-clustering methods, such as agglomerative clustering algorithms, that assign each entity mention to exactly one cluster. Moreover, the high dimensional input space of VAE is better equipped to encode variations in the observed surface forms of different entity/relation mentions.

The generative process of VaDE is described as follows. Assuming that there are K clusters, an observed instance $x \in \mathbb{R}^D$ is generated as,

1. Choose a cluster $c \sim \text{Cat}(\pi)$, i.e. a categorical distribution parametrized by probability vector π.

2. Choose a latent vector $z \sim \mathcal{N}(\mu_c, \sigma_c^2\mathbf{I})$ i.e. sample z from a multi-variate Gaussian distribution parametrized by mean μ_c and diagonal covariance $\sigma_c^2\mathbf{I}$.

3. Compute $[\mu_x; \log \sigma_x^2] = f_\theta(z)$ where f_θ corresponds to a neural network parametrized by θ, and z is obtained from the previous step.

4. Finally, choose a sample $x \sim \mathcal{N}(\mu_x, \sigma_x^2\mathbf{I})$ i.e. sample x from a multi-variate Gaussian distribution parametrized by mean μ_x and diagonal covariance $\sigma_x^2\mathbf{I}$.

where π_k is the prior probability for cluster k, $\pi \in \mathbb{R}_+^K$ and $\sum_{k=1}^K \pi_k = 1$. We make the same assumptions as made by (Jiang et al., 2017a), and assume the variational posterior $q(z, c|x)$ to be a mean field distribution, and factorize it as:

$$q(z, c|x) = q(z|x)q(c|x) \tag{1}$$

We describe below the *inner workings* of CUVA with respect to the *head* Noun Phrase, or the leftmost vertical structure in Fig. 1. An analogous description follows for the *tail* Noun Phrase, and the Relation Phrase as well.

Encoder: Fig. 2 illustrates the Encoder block graphically. A Noun Phrase h, is fed as input to the Encoder block, which consists of: *a*) An embedding lookup table, *b*) A *two* layer fully connected neural network g_ϕ^E (g_ϕ^R for R-VAE) with *tanh* non–linearity, and *c*) Two linear layers in parallel. .

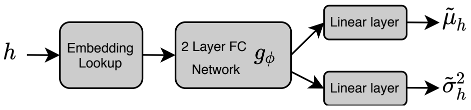

Figure 2: An Encoder block. CUVA uses separate fully connected networks g_ϕ^E and g_ϕ^R for E-VAE and R-VAE.

The Encoder block is used to model $q(z|h)$ i.e. the variational posterior probability of the latent representation z given input representation h, via the following equations,

$$[\tilde{\mu}_h; \log \tilde{\sigma}_h^2] = g_\phi^E(h) \tag{2}$$

$$q(z|h) = \mathcal{N}(z; \tilde{\mu}_h, \tilde{\sigma}_h^2 \mathbf{I}) \tag{3}$$

After the parameters $\tilde{\mu}_h, \tilde{\sigma}_h$ for the variational posterior $q(z|h)$ have been calculated, we use the reparametrization trick (Kingma and Welling, 2014) to sample z_1 as follows,

$$z_1 = \tilde{\mu}_h + \tilde{\sigma}_h \circ \epsilon \tag{4}$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ (i.e. a standard normal distribution) and $\circ$ denotes element-wise multiplication.

Decoder: Given z_1, the *decoding* phase continues through the Decoder block, as illustrated in Fig. 3, and via the following equations,

$$[\tilde{\mu}; \log \tilde{\sigma}^2] = f_\theta^E(z_1) \tag{5}$$

$$h' \sim \mathcal{N}(\tilde{\mu}, \tilde{\sigma}^2 \mathbf{I}) \tag{6}$$

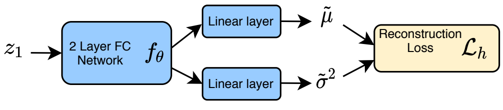

Figure 3: An Decoder block. CUVA uses separate fully connected networks f_θ^E and f_θ^R for E-VAE and R-VAE.

Following (Jiang et al., 2017a), the variational posterior $q(c|h)$ i.e. the probability of the NP h belonging to cluster c is calculated as:

$$q(c|h) = \frac{p(c)p(z|c)}{\sum_{c'=1}^{K} p(c')p(z|c')} \tag{7}$$

In practice, we use z_1 obtained from Equation 4 in place of z (in Equation 7), and calculate a vector of assignment probability for an input h.

During *inference* phase, h is assigned to a cluster having the highest probability, i.e. cluster assignment (in Fig. 1) occurs via a *winners-take-all* strategy.

3.2 The KGE Module

The *motivation* behind using a Knowledge Graph Embedding (KGE) module is to encode the structural information present within the Open KG. This module is responsible for the joint learning between the latent representations for entities and relations (See Figure 1) and is described as follows.

Given a triple mention (h, r, t) belonging to an Open KG, we use Equation 7 to obtain a vector of cluster assignment probabilities c^h, c^r and c^t for the NPs and RPs respectively. As the next step, we choose a base $\tau > 0$ and employ a soft argmax function on probability vectors c^h, c^r and c^t as follows,

$$\sigma(c_\alpha) = \frac{e^{\tau c_i^\alpha}}{\sum_{j=1}^{K} e^{\tau c_j^\alpha}}, \text{for } i = 1, \ldots, K \tag{8}$$

where $\alpha \in \{h, r, t\}$ and K denotes the number of clusters.

Choosing a large value of τ ensures that the resulting vectors v_h, v_t and v_r obtained from Equation 8 are one-hot in nature, and indicate the most probable cluster ids for the NPs h, t and RP r respectively. For all our experiments, we choose $\tau = 1e5$. In short, Equation 8 is a differentiable approximation to the non-differentiable argmax function.

Given that, we now know the most probable cluster ids for a triple mention (h, r, t), we build the *entity* and *relation* representations of these mentions, namely e_h, e_t and e_r as,

$$e_h = v_h M_E \quad e_t = v_t M_E \quad e_r = v_r M_R \tag{9}$$

where M_E, M_R represent matrices containing *mean* vectors (stacked across rows) for each of the K_E and K_R Gaussians present in E-VAE and R-VAE respectively. Here, K_E and K_R (Fig. 1) are hyper-parameters for CUVA.

Once, we have the *entity* and *relation* representations, we use HolE as described in Nickel et al. (2016) as our choice of KGE algorithm for CUVA.

3.3 Side Information

Noun and Relation Phrases present within an Open KG can be often tagged with relevant side information extracted from the context sentence in which

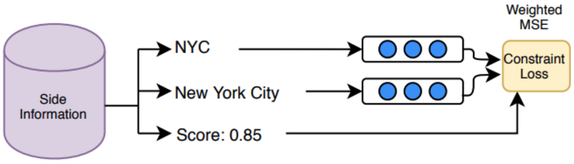

Figure 4: Encoding Side Information as a *loss* measure.

the triple appears. We use the same side information (i.e. a list of equivalent mention pairs) as CESI (Vashishth et al., 2018a). These side information tuples are obtained via the following sources/strategies: Entity Linking, PPDB (ParaPhrase DataBase), IDF Token Overlap, and Morph Normalization. Each source generates a list of equivalent mention pairs along with a score per pair. A description of these sources together with their associated scoring procedures is provided in Section A of the Appendix.

Let's consider the example of an *equivalent* mention pair *(NYC, New York City)* as shown in Fig. 4 to illustrate the use of side information as a constraint in CUVA. We first perform an embedding lookup for the mentions *NYC* and *New York City* and then compute a Mean Squared Error (MSE) value weighted by its plausibility score. The MSE value indicates how far CUVA is from satisfying all the constraints represented as equivalent mention pairs. Finally, we sum up the *weighted* MSE values for all equivalent mention pairs, which comprises our Side Information Loss $\mathcal{L}_{\text{SI}}$ in Fig. 1.

4 Evaluation

The CANONICALIZATION task is *inherently* unsupervised, i.e. we are not given any manually annotated data for training. With this in mind, we train the CUVA model according to the procedure described in Section B of the Appendix and then evaluate our approach on the *Entity Canonicalization* task only. We do not include quantitative evaluations on the *Relation Canonicalization* task, as none of the benchmarks described below have ground-truth annotations for canonicalizing relations, leaving the creation of a dataset for relation clustering as an interesting future work.

4.1 Benchmarks

For comparing the performance of CUVA against the existing state of the art approaches, we use the Base and Ambiguous datasets introduced by (Galárraga et al., 2014a), and ReVerb45K dataset from (Vashishth et al., 2018a).

Datasets	Gold NP Clusters	NPs	RPs	Triples
Base	150	290	3K	9K
Ambiguous	446	717	11K	37K
ReVerb45K	7.5K	15.5K	22K	45K
CANONICNELL	1.4K	8.7K	139	20K

Table 1: Details of datasets used. CANONICNELL is the new dataset that we introduce in this paper.

In addition, we introduce a *new* dataset called CANONICNELL, which we built by using the 165th iteration snapshot of NELL, i.e. Never-Ending Language Learner (Carlson et al., 2010) system. We created CANONICNELL to build a dataset whose provenance is not related to ReVerb Open KB, unlike the datasets mentioned above.

Building CANONICNELL. The CANONICNELL dataset is built via an automated strategy as follows. The above snapshot of NELL aka NELL165, contains accumulated knowledge as a list of (subject, relation, object) triples. For building CANONICNELL, we use the data artifact generated by (Pujara et al., 2013) which marks co-referent entities within NELL165 triples, together with a soft-truth value per entity pair. We filter out all pairs having a score *less* than 0.25, and view the remaining pairs as undirected edges in a graph. To this graph, we apply a depth-first-search to obtain a set of connected components, which we refer to as the set of Gold Clusters. Next, we filter through the list of NELL165 triples and keep only those whose either *head* or *tail* entity is present within the set of *Gold Clusters*. These triples together with the Gold Clusters obtained previously, form our newly proposed CANONICNELL dataset.

Table 1 shows the dataset statistics for all the benchmarks, wherein the split into test and validation folds for Base, Ambiguous and ReVerb45K datasets is already given by Vashishth et al. (2018a)[2]. This task is *unsupervised* in nature, hence we do not possess any training data. For CANONICNELL, we did a random 80:20 split of the triples into validation and test folds. For all methods, grid search over the hyper-parameter space using the validation set is performed, and results corresponding to the best-performing settings are reported on the test set. Following Galárraga et al. (2014a), we use the macro, micro, and pair F1 scores for evaluations.

[2] https://github.com/mallabiisc/cesi/

		Base Dataset			Ambiguous Dataset			ReVerb45K		
	SI	Macro	Micro	Pair	Macro	Micro	Pair	Macro	Micro	Pair
Galárraga-IDF†	No	0.948	0.979	0.983	**0.679**	0.829	0.793	**0.716**	0.508	0.005
GloVe+HAC†	No	0.957	0.972	0.911	0.659	0.899	0.901	0.565	0.829	0.753
GloVe+HAC+SI	Yes	0.972	0.998	0.999	0.665	0.898	0.764	0.666	0.847	0.708
HolE (GloVe)†	No	0.752	0.936	0.893	0.539	0.854	0.767	0.335	0.758	0.510
CESI†	Yes	**0.982**	0.998	**0.999**	0.662	**0.924**	0.919	0.627	0.844	0.819
CUVA	Yes	**0.982**	**0.999**	**0.999**	0.674	**0.924**	**0.92**	0.661	**0.867**	**0.855**

Table 2: Macro, Micro and Pair F1 results on the *Entity Canonicalization* task for *head entity mentions*. Rows marked with a † are from (Vashishth et al., 2018a). CESI, the existing state-of-the-art approach is identical to HolE (GloVe) equipped with the Side Information. SI indicates whether an approach uses Side Information or not. Refer to Sec C of the Appendix for the hyperparameter descriptions.

	SI	Macro	Micro	Pair	Mean
CESI	Yes	0.749	0.828	0.763	0.780
CUVA	Yes	**0.755**	**0.845**	**0.81**	**0.803**

Table 3: Macro, Micro and Pair F1 results on the *Entity Canonicalization* task for *all entity mentions* on ReVerb45K. Both CESI and CUVA use the same hyperparameter settings as in Table 2.

Following Vashishth et al. (2018a), we use the Side Information as mentioned in Section 3.3 for canonicalizing NPs and RPs for the Base, Ambiguous, and ReVerb45K datasets. For canonicalizing NPs on CANONICNELL, we use IDF Token Overlap as the only strategy to generate Side Information. This strategy is an inherent property of the dataset and needs no external resources (Section A). Moreover, for CANONICNELL we do not canonicalize the RPs, since they are already unique.

Finally, a detailed description of the range of values tried per hyperparameter, and the final values used within CUVA is provided in Section C.

4.2 Results

The existing state of the art model CESI (Vashishth et al., 2018a) evaluates on the Entity Canonicalization task using *head entity mentions* only[2]. To be comparable, we first evaluate on the head entity mentions only and illustrate our results in Table 2. Table 3 illustrates the results when evaluated for all entity mentions on Reverb45K.

The *first* line in Table 2, i.e. Galárraga-IDF (Galárraga et al., 2014a) depicts the performance of a feature-based method on this task. This approach is more likely to put two NPs together if they share a token with a high IDF value. The *second* row in Table 2, i.e., GloVe+HAC uses a pretrained GloVe model (Pennington et al., 2014a) to first build embeddings for entity mentions and then uses a HAC algorithm for clustering. For multi token phrases, GloVe embeddings for tokens are averaged together. GloVe captures the semantics of NPs and does not rely on its surface form, thus performing well across all the datasets. The *third* row augments GloVe+HAC by first initializing with pretrained GloVe vectors, followed by an optimization step wherein the Side Information (Section 3.3) loss objective is minimized, and finally clustering via the HAC algorithm. The *fourth* row, i.e. HolE (GloVe) uses the HolE Knowledge Graph Embedding model (initialized with pretrained GloVe vectors) to learn unique embeddings for NPs and RPs, followed by clustering using a HAC algorithm. It captures structural information with the KG and is an effective approach for NP Canonicalization.

The current state of the art, i.e. CESI (Vashishth et al., 2018a) extends the HolE (GloVe) approach, by adding Side Information (Section 3.3) as an additional loss objective to be minimized. Looking at the results, it is clear that the addition of Side Information provides a *significant* boost in performance for this task.

The final row in Table 2 illustrates CUVA's performance on this task, which is an original contribution of our work. We observe that CUVA outperforms CESI on ReVerb45K and achieves the new state-of-the-art (SOTA). The improvement in the Mean F1 value, i.e., average over Macro, Micro, and Pair F1, over CESI is statistically significant with *p value* being less than 1e-3.

On the Ambiguous dataset, CUVA achieves a 1.2% improvement over CESI on the Macro F1

	SI	Macro	Micro	Pair
FastText+HAC	No	0.725	0.798	0.219
GloVe+HAC	No	0.747	0.811	0.280
CESI	Yes	0.749	0.817	0.307
CUVA	Yes	**0.775**	**0.826**	**0.363**

Table 4: Macro, Micro and Pair F1 results on the *Entity Canonicalization* task for the CANONICNELL dataset. Refer to Section C of the Appendix for the hyperparameters used in these experiments. SI indicates whether an approach uses Side Information or not.

metric, with the Micro and Pair F1 metrics achieving identical performance as CESI. Finally, on the Base dataset, CUVA achieves identical performance as CESI. Moreover, the results in Table 3 also show a similar trend when evaluated on all entity mentions, i.e., both head and tail NPs belonging to Reverb45K.

Table 4 shows the results for the Entity Canonicalization task when evaluated on the CANONICNELL dataset. The first two rows correspond to approaches that use pretrained FastText (Mikolov et al., 2018) and GloVe models to build unique embeddings for NPs and then use HAC to generate clusters. Moreover, in the absence of contextual information for the CANONICNELL triples, both CESI and CUVA use IDF Token Overlap as the only source of Side Information. Moreover, from Table 4, it is clear that CUVA achieves the new state of the art result on this benchmark as well.

5 Qualitative Analysis

Table 5 illustrates the output of our system for canonicalizing NPs and RPs on ReVerb45K. The top block corresponds to *six* NP clusters, one per line. The algorithm is able to correctly group *kodagu* and *coorg* (different name of the same district in India), despite having completely different surface forms. However, a common mistake that our proposed system makes is depicted in row *five*, i.e., four different people each having the same name *bill* are clustered together. This error can be mitigated by keeping track of the *type* information (Dash et al., 2020) of each NP for disambiguation.

The bottom *four* rows in Table 5 correspond to *four* RP clusters. While the equivalence of RPs is captured in the first two rows of the bottom block, the *final* two rows highlight a potential issue involving negations and antonyms, i.e. *rank below* and *be rank above* have opposite meanings. We leave the resolution of this issue as future work.

Predicted Clusters for ReVerb45K
{utc, coordinate universal time, universal coordinate time} ✔
{justice alito, samuel alito, sam alito, alito} ✔
{kodagu, coorg} ✔
{johnny storm, human torch} ✔
{bill cosby, bill maher, bill doolin, bill nye} ✖
{toyota, honda, toyota motor corporation} ✖
{be associate with, have be affiliate to, be now associate with} ✔
{lead a march on, lead the assault on} ✔
{be far behind, be not far behind, be way behind, be close behind, be seat behind, be firmly entrench in } ✖
{rank below, rank just below, be rank above} ✖

Table 5: Examples of NP(*top*) and RP(*bottom*) clusters predicted by CUVA on ReVerb45K.

	Macro	Micro	Pair	Mean
RoBERTa+HAC	0.448	0.804	0.776	0.676
BERT+HAC	0.586	0.839	0.822	0.749
ERNIE+HAC	0.591	0.842	0.825	0.753
CUVA	**0.661**	**0.867**	**0.855**	**0.794**

Table 6: Macro, Micro and Pair F1 results for comparison with pretrained language models on the *Entity Canonicalization* task for *head entity mentions* on ReVerb45K.

6 Further Analysis

In this section, we analyze CUVA under three different configurations. Section 6.1 compares how CUVA performs against pretrained language models. Section 6.2 analyzes the effect of ablating components from our proposed network architecture. Finally, Section 6.3 demonstrates the effectiveness of a joint learning approach over a pipeline-based strategy using the same network architecture.

6.1 Comparison with Pretrained LMs

In this section, we investigate how CUVA fares against pretrained language models. Table 6 illustrates the results when evaluated on ReVerb45K.

Following the observations of (Liu et al., 2019; Tenney et al., 2019), we use the lower layers (layers *one* through *six*) of a pretrained BERT, RoBERTa and a knowledge graph enhanced ERNIE (Zhang et al., 2019) base model via the HuggingFace Transformers library (Wolf et al., 2020). For building static representation for each entity mentions, we use a mean pooling strategy to aggregate the contextualized representations. The entity mention representations are finally clustered using HAC.

Empirically, we found layer *one* to work best for all the language models introduced above. Furthermore, RoBERTa performs *worse* out of the three when comparing the derived static embeddings on this task. In comparison, CUVA performs significantly better, i.e. +4.1%, +4.5% and +11.8% improvement on the *average* of Macro, Micro, and Pair F1 values, when compared against ERNIE+HAC, BERT+HAC and RoBERTa+HAC respectively.

6.2 Structural Ablations

Table 7 illustrates the ablation experiments performed on the Entity Canonicalization task for the CanonicNell dataset. The *first* row corresponds to CUVA model used to obtain state-of-the-art results, as reported in Table 4. Removing the *hidden* layer out of CUVA's encoder and decoder network, yields the results in the *second* row of the table. The *final* row reports the performance of CUVA without the KGE Module.

From the results, it is clear that adding an *hidden* layer to the VAE certainly improves CUVA's performance. Moreover, we find that removing the KGE Module drops the Pair F1 value *significantly* by 8.4%, with a statistically *insignificant* increase in the Macro and Micro F1 values. This *drop* in Pair F1 is due to a drop in the pairwise precision, i.e., from 0.379 with KGE to 0.229 without KGE.

Pairwise precision measures the quality of a set of clusters as the ratio of number of hits to the total possible allowed pairs, wherein a pair of NPs produce a hit if they refer to the same entity. Therefore, using a KGE module causes CUVA to generate a higher hit ratio, and in turn *supports* our hypothesis that a KGE Module helps to better disambiguate entity clusters by considering the context given by the relations, and is therefore necessary.

6.3 Effectiveness of Joint Learning

CUVA models the Canonicalization task via a latent variable generative model and approximates the likelihood of an observed Open KG triple via a variational inference approach. Under this method, the probability of an NP (or RP) belonging to a latent cluster is entangled with both the representations of the observed mentions and the representations of the latent, and consequently, affects the likelihood of an observed triple in a joint manner. This is relevant because it allows gradients to update both the mention embeddings and soft cluster assignments jointly, thereby effectively learning from one another.

Approaches	Macro	Micro	Pair
CUVA	0.775	0.826	0.363
Without the Hidden Layer	0.758	0.809	0.253
Without the KGE Module	0.782	0.829	0.279

Table 7: Ablation tests on the Entity Canonicalization task showing Macro, Micro and Pair F1 results for the CanonicNell dataset. All approaches here use the same hyperparameters as in Table 4.

Approaches	Macro F1	Micro F1	Pair F1
CUVA	0.661	0.867	0.855
VAE+HAC	0.545	0.862	0.777

Table 8: Ablation tests illustrating that our proposed approach for *joint learning* of mention representations and cluster assignments performs better than a *pipeline* approach. The evaluations have been done on the Entity Canonicalization task (*head mentions only*) for the ReVerb45K dataset.

Table 8 empirically demonstrates this relevance, i.e. benefits of joint learning over a pipeline approach, while using the same network architecture. In this study, the experiments have been done on the Entity Canonicalization task (*head mentions* only) for the ReVerb45K dataset. In addition to CUVA, we build a second model following a *pipeline* approach, which we refer to as VAE+HAC. This model first uses the same architecture as CUVA for learning mention representations, and in a subsequent independent step, uses a hierarchical agglomerative clustering step to cluster the mentions together. The results indicate that a joint approach outperforms a pipeline-based strategy used by existing state-of-the-art models, such as CESI.

7 Conclusion

In this paper, we introduced CUVA, a novel neural architecture to canonicalize Noun Phrases and Relation Phrases within an Open KG. We argued that CUVA learns unique mention embeddings and cluster assignments in a *joint* fashion, compared to a pipeline strategy followed by the current state of the art methods. Moreover, we also introduced CanonicNell, a new dataset for Entity Canonicalization. An evaluation over four benchmarks demonstrates the effectiveness of CUVA over state of the art baselines.

References

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354, Beijing, China. Association for Computational Linguistics.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 2670–2676.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

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*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*. AAAI Press.

Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, Raffaele Perego, and Salvatore Trani. 2014. Dexter 2.0 - an open source tool for semantically enriching data. In *Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014*, volume 1272 of *CEUR Workshop Proceedings*, pages 417–420. CEUR-WS.org.

Sarthak Dash, Md. Faisal Mahbub Chowdhury, Alfio Gliozzo, Nandana Mihindukulasooriya, and Nicolas Rodolfo Fauceglia. 2020. Hypernym detection using strict partial order networks. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7626–7633. AAAI Press.

D. Defays. 1977. An efficient algorithm for a complete link method. *Comput. J.*, 20(4):364–366.

Claudio Delli Bovi, Luis Espinosa-Anke, and Roberto Navigli. 2015. Knowledge base unification via sense embeddings and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 726–736, Lisbon, Portugal. Association for Computational Linguistics.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011a. Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011b. Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Luis Galárraga, Geremy Heitz, Kevin Murphy, and Fabian M. Suchanek. 2014a. Canonicalizing open knowledge bases. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014, Shanghai, China, November 3-7, 2014*, pages 1679–1688. ACM.

Luis Galárraga, Geremy Heitz, Kevin Murphy, and Fabian M. Suchanek. 2014b. Canonicalizing open knowledge bases. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014, Shanghai, China, November 3-7, 2014*, pages 1679–1688. ACM.

Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, pages 413–422. International World Wide Web Conferences Steering Committee / ACM.

Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. 2017a. Variational deep embedding: An unsupervised and generative approach to clustering. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1965–1972. ijcai.org.

Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. 2017b. Variational deep embedding: An unsupervised and generative approach to clustering. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1965–1972. ijcai.org.

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

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Jayant Krishnamurthy and Tom Mitchell. 2011. Which noun phrases denote which concepts? In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 570–580, Portland, Oregon, USA. Association for Computational Linguistics.

Thomas Lin, Mausam, and Oren Etzioni. 2012. Entity linking at web scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction, AKBC-WEKEX@NAACL-HLT 2012, Montrèal, Canada, June 7-8, 2012*, pages 84–88. Association for Computational Linguistics.

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

Stuart P. Lloyd. 1982. Least squares quantization in PCM. *IEEE Trans. Inf. Theory*, 28(2):129–136.

Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534, Jeju Island, Korea. Association for Computational Linguistics.

Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural variational inference for text processing. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1727–1736. JMLR.org.

Tomás Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association (ELRA).

Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio. 2016. Holographic embeddings of knowledge graphs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 1955–1961. AAAI Press.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 425–430, Beijing, China. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014a. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014b. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Jay Pujara, Hui Miao, Lise Getoor, and William W. Cohen. 2013. Knowledge graph identification. In *The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part I*, volume 8218 of *Lecture Notes in Computer Science*, pages 542–557. Springer.

Valentin I. Spitkovsky and Angel X. Chang. 2012. A cross-lingual dictionary for english wikipedia concepts. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012, Istanbul, Turkey, May 23-25, 2012*, pages 3168–3175. European Language Resources Association (ELRA).

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 885–895, New Orleans, Louisiana. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Shikhar Vashishth, Prince Jain, and Partha P. Talukdar. 2018a. CESI: canonicalizing open knowledge bases using embeddings and side information. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1317–1327. ACM.

Shikhar Vashishth, Prince Jain, and Partha P. Talukdar. 2018b. CESI: canonicalizing open knowledge bases using embeddings and side information. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1317–1327. ACM.

Aaron Steven White, Drew Reisinger, Keisuke Sakaguchi, Tim Vieira, Sheng Zhang, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. 2016. Universal decompositional semantics on Universal Dependencies. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1713–1723, Austin, Texas. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Tien-Hsuan Wu, Zhiyong Wu, Ben Kao, and Pengcheng Yin. 2018. Towards practical open knowledge base canonicalization. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 883–892. ACM.

Alexander Yates and Oren Etzioni. 2009. Unsupervised methods for determining object and relation synonyms on the web. *J. Artif. Intell. Res.*, 34:255–296.

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

A Side Information

Following CESI (Vashishth et al., 2018b), we use the following five sources of side information, which are described as follows:

- **Entity Linking**: Given unstructured text, from which the triple was extracted, we use Stanford CoreNLP entity linker (Spitkovsky and Chang, 2012) to map Noun Phrases (NPs) to Wikipedia Entities. If two NPs are linked to the same Wikipedia entity, we assume them to be equivalent as per this information.
- **PPDB Information**: We follow the same strategy as (Vashishth et al., 2018b) and modify the PPDB 2.0 (Pavlick et al., 2015) collection into a set of clusters. If two NPs (or RPs) belong to the same cluster, then they are treated as equivalent.
- **IDF Token Overlap**: In (Galárraga et al., 2014b), IDF Token Overlap was found to be the most effective feature for canonicalization. For example, it is very likely that *William Shakespeare* and *Shakespeare* refer to the same entity, or in other words, Noun Phrases (NPs) or Relation Phrases (RPs) sharing infrequent terms are more likely to refer to the same entity (or relation). An overlap score for every NP (or RP) pair is calculated as per the formula provided in (Vashishth et al., 2018b), and we keep only those pairs with scores beyond a particular *threshold*.
- **Morph Normalization**: We use multiple morphological normalization operations, as used in (Fader et al., 2011b) for finding out equivalent NPs.

We use the following strategy to calculate the *plausibility scores* for the mention pairs generated by each of the *five* aforementioned sources of Side Information. Mention pairs identified by IDF Token Overlap follow the same *scoring* strategy as mentioned before, whereas mention pairs identified by WordNet (with Word-sense disambiguation) and Morphological normalizations get a score of *one*.

The remaining sources, i.e. Entity Linking and PPDB, tend to group the NP and RP mentions into clusters. Being empirical in nature, these approaches are *likely* to introduce errors in their results, for e.g. due to incorrect disambiguation, and can cause some of the generated clusters to overlap.

Working with such a set of potentially overlapping clusters, we make an *observation* that, if a particular *mention* belongs to more than one cluster, then it is likely to be ambiguous, and therefore should have a low equivalence score with other members of the same cluster. Therefore, we score two mentions p and q belonging to the same cluster $\mathcal{C}$ as,

$$S_{\mathcal{C}}(p,q) = \frac{1}{|\mathcal{C}|^2} e^{2-(\eta(p)+\eta(q))}$$

where e denotes the exponential function, $\eta(x)$ denotes the number of clusters containing x, and $|\mathcal{C}|$ denotes the cluster size. The scaling factor of $1/|\mathcal{C}|^2$ favors clusters of smaller size, since for the CANONICALIZATION task, ideal cluster sizes are expected to be small.

B Training Strategy

In this section, we describe our strategy for training the CUVA model. Let $\mathcal{E}, \mathcal{R}$ denote the entity and relation vocabulary for an Open KG. Unless otherwise specified, all *trainable* CUVA parameters are *randomly* initialized. We train the model in *three* stages, as follows:

B.1 Initializing Mixture of Gaussians

We use the *pretrained* 100-dimensional GloVe vectors (Pennington et al., 2014b) for embedding matrices $\mathcal{E}_g$ and $\mathcal{R}_g$ corresponding to the vocabulary $\mathcal{E}$ and $\mathcal{R}$ respectively.

The embeddings for multi-token phrases are calculated by averaging GloVe vectors for each token. This step can be done in one of two ways, *a*) Normalize individual GloVe token vectors and then average them, or *b*) Average individual GloVe token vectors without Normalizing. In the absence of any other information, we evaluate CUVA on the validation fold of each of the benchmark datasets, as shown in Table 9. For each dataset, we mark the embedding initialization strategy that yields the best performance, and then use it to evaluate our model on the test fold of the corresponding benchmark datasets (as illustrated in the main paper).

Based on the results from Table 9, we use the *Without Normalization* strategy for Ambiguous dataset, whereas for ReVerb45K, we use the *With Normalization* strategy. For the Base dataset, both strategies yield the same results and therefore we randomly choose the *With Normalization* strategy and use it while evaluating on the test fold. Furthermore, we choose the *With Normalization* strategy

	With Normalization			Without Normalization		
Validation fold	Macro	Micro	Pair	Macro	Micro	Pair
Base	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
Ambiguous	0.811	0.966	0.964	**0.823**	**0.976**	**0.966**
ReVerb45K	**0.728**	**0.906**	**0.953**	0.721	0.901	0.951

Table 9: Comparison of *two* initialization strategies when used for initializing mixture of gaussians for running CUVA on the Entity Canonicalization task for *head entity mentions* on the *Validation fold* of the benchmark datasets. The *With Normalization* strategy builds GloVe embeddings for multi-token NPs by averaging unit L2 normalized GloVe vectors for individual tokens, whereas the *Without Normalization* strategy simply averages GloVe vectors for individual tokens to build the embeddings for multi-token NPs. See Section B.1 for more details.

for the CANONICNELL dataset as well.

For the CANONICALIZATION task, the cluster sizes will be likely small, and in turn, we get a large number of clusters. The average-case time complexity per iteration of k-Means using Lloyd's algorithm (Lloyd, 1982) is $O(nk)$, where n is the number of samples. However, for our case, as k is comparable to n, the average time complexity becomes $O(n^2)$ similar to the Hierarchical Agglomerative Clustering (HAC) method with complete linkage criterion (Defays, 1977). Though both methods have the same time complexity, we use HAC as our clustering method as we observe that it gives a better performance empirically. We cover the empirical comparison between both methods of initialization, i.e. HAC and KMeans in Section D of this Appendix.

We run HAC separately over $\mathcal{E}_g$ for NPs, and $\mathcal{R}_g$ over RPs. We use two different thresholds θ_E for entities, and θ_R for relations to convert the output dendrograms from HAC into *flat* clusters. Using these clusters, we compute within-cluster means and variances to initialize the *means* and the *variances* of the Gaussians for both E-VAE and R-VAE respectively. Note that, the choice of θ_E and θ_R sets the values for the number of mixtures K_E and K_R used in the next stage.

B.2 Two-step training procedure

We train CUVA in *two* independent steps. Our training strategy is similar to (Miao et al., 2016) where they train the encoder and decoder of the VAE alternatively rather than simultaneously. In the first step, we train the encoder in both E-VAE and R-VAE while keeping the decoder fixed. Then, in the second step, we keep the encoder fixed and only train the decoder.

Encoder training: We train the *Encoder* for both E-VAE and R-VAE by using the labels generated via the HAC algorithm (during initialization of the mixture of gaussians) as a source of weak supervision. Specifically, for a given triple (h, r, t), we compute:

- Negative log likelihood (NLL) loss $\mathcal{L}_h$ calculated using the predicted cluster assignment probability vector for h and the cluster label for h.
- NLL values $\mathcal{L}_r, \mathcal{L}_t$ for r, t computed in a similar manner.
- L1 Regularizer values using the *Encoder* parameters for E-VAE and R-VAE, denoted by $\mathcal{L}_{\text{REG1}}$.
- Side Information Loss $\mathcal{L}_{\text{SI}}$ applicable between any two equivalent NPs (or RPs). See Figure 1.

The *overall* loss function for the *first* step is therefore,

$$\mathcal{J} = \sum_{(h,r,t)\in\mathcal{T}} \mathcal{L}_h + \mathcal{L}_r + \mathcal{L}_t + \lambda\mathcal{L}_{\text{REG1}} + \mathcal{L}_{\text{SI}}$$

We train the *Encoder* for a maximum of T_e epochs, and then proceed to the *second* step.

Using labels generated by the HAC algorithm as a source of weak supervision introduces noise and sets an upper limit to how much CUVA can learn. However, we also use side information during the Encoder training procedure, which helps CUVA fix the errors introduced by HAC, thus resulting in an *improved* performance. This behavior is empirically demonstrated by comparing GloVe+HAC and CUVA approaches on the ReVerb45K dataset in the main paper.

Decoder training: In this step, we train the *decoder* only, and keep the *encoder* fixed. The cluster parameters and the embedding lookup table are also updated. The decoder is trained by minimizing the following *loss* values:

- The evidence lower bound (ELBO) loss $\mathcal{L}_{\text{ELBO}}^{E}$ for E-VAE and $\mathcal{L}_{\text{ELBO}}^{R}$ for R-VAE respectively, with the *decoder* being a multivariate Gaussian with a diagonal covariance structure. The ELBO loss breaks into two parts namely, the Reconstruction Loss, and the KL divergence between the variational posterior and the prior. The expressions for ELBO loss are based on (Jiang et al., 2017b).
- The KGE Module loss $\mathcal{L}_{\text{KGE}}$ and the Side Information Loss $\mathcal{L}_{\text{SI}}$ (Refer to Figure 1).
- L1 Regularizer loss values ($\mathcal{L}_{\text{REG2}}$) using the *Decoder* parameters for E-VAE and R-VAE.

The *combined* loss function for the *second* step is:

$$\mathcal{J} = \mathcal{L}_{\text{ELBO}}^{E} + \mathcal{L}_{\text{ELBO}}^{R} + \mathcal{L}_{\text{KGE}} + \mathcal{L}_{\text{SI}} + \lambda\mathcal{L}_{\text{REG2}}$$

where λ corresponds to the *weight* value for the regularizer, a hyper-parameter set to 0.001. The decoder is trained for a maximum of T_d epochs.

The motivation behind using a two-step training strategy for the VAEs is to prevent the decoder from ignoring latent representations z and learning directly from the input data (Bowman et al., 2016). Once the encoder has been trained in the first step, we keep the encoder weights fixed for the second step. This forces the decoder to learn only from the *latent* representations, and not from the input data. Note that the KGE loss $\mathcal{L}_{\text{KGE}}$ is not used in Step one, since it causes the model to diverge in practice.

C Hyperparameters

In this section, we discuss the grid search for hyperparameters and present the final hyperparameters used.

C.1 Grid Search Details

The search space used to obtain the best performing hyper-parameters for our experiments is described as follows: We calculate the *threshold* cutoff for HAC based initializations using the validation fold via a two-step approach. In the *first* step, we use a search space of $[0.2, 1.0)$ in steps of 0.1. In the *final* step, we take the best cutoff value c from the previous step and construct a new search space $[c - 0.1, c + 0.1]$ with a step size of 0.01. Finally, we take the best performing threshold cutoff value from the previous step and use it to evaluate CUVA models on the test set.

For choosing the *threshold* cutoff for the IDF Token Overlap strategy in regards to Entity Side Information, we used a search space of $[0.2, 0.8]$ in increments of 0.1 for all the datasets. In comparison, we chose 0.9 as a cutoff for the IDF Token Overlap strategy in regards to Relation Side Information (wherever applicable), without any search as it already produced a decent number of relation pairs and manual inspection of a sample indicated good quality. Finally, as to the choice of the latent space dimensions for the VAE, we employed a grid search over $\{50, 100, 200\}$ dimensions.

C.2 Final Hyperparameters used

We use the following hyperparameter values in our experiments.

Common hyperparams. The fully connected layers in the Encoder section of the VAEs have embedding dimensions of 768, 384, and 100, whereas the Decoder sections have the same dimensions, but in reverse order. Both Encoder and Decoder use *tanh* nonlinearities. A learning rate of 1e-3 and 1e-4 together with Adam optimizer (Kingma and Ba, 2015) is used in steps one and two during our proposed two-step training procedure. L1 regularization with a regularizer weight of 1e-3 is used. A batch size of 50 is used for training, whereas for evaluation, we use a batch size of 5. Moreover, we use 20 random negative samples per positive sample, while calculating the loss function pertaining to the HolE algorithm. The GloVe vectors used for initializing the Gaussian Mixture models are obtained from `http://nlp.stanford.edu/data/GloVe.6B.zip`.

For Base, Ambiguous and ReVerb45K datasets, we use a threshold of 0.4 for entities and 0.9 for relations regarding the IDF Token Overlap strategy for scoring Side Information pairs, i.e. pairs whose scores are less than these cutoff values, are discarded. For CanonicNell we employ a threshold of 0.5 concerning the IDF Token Overlap strategy for scoring Entity Side Information pairs. Furthermore, the relations within CanonicNell are unique, therefore they are treated as singleton clus-

Params	Base	Ambiguous	ReVerb45K	CanonicNell
θ_E	0.53	0.3	0.4	0.21
θ_R	0.43	0.5	0.37	N/A
K_E	1021	5013	12965	6625
K_R	102	625	1076	N/A
T_e	50	50	50	50
T_d	300	300	300	100
Seed	42	57	55	10

Table 10: Final dataset specific hyperparameters used for training and evaluating CUVA models on the test fold of these benchmark datasets. Here, θ_E and θ_R denote the threshold cutoff used during HAC based initializations. Setting the values of θ_E and θ_R sets the values for the number of mixtures K_E and K_R used in the E-VAE and R-VAE respectively. See Section B.1 for a detailed description on the notations used.

ters for the experiments.

Dataset specific hyperparameters. The dataset specific hyperparameters are illustrated in Table 10. The first six rows corresponds to hyperparameters related to our proposed CUVA model, whereas the final row pertains to the Seed values (for reproducibility purposes) used for evaluating CUVA models on the test fold of the benchmark datasets.

Moreover, all experiments are implemented in PyTorch v1.4.0 using a single Intel x86 CPU and one NVIDIA v100 GPU, with a max of 16GB RAM.

D Other Ablation Experiments

In this section, we describe additional experiments to analyze the performance of our proposed CUVA model.

Table 11 illustrates the performance of CUVA while varying the strategies on the choice of initializations for the Gaussian Mixture model and Knowledge Graph Embedding. While our proposed instantiation of CUVA, i.e. Row *two*, uses HAC clustered GloVe vectors for initializations and HolE for Knowledge Graph Embedding, it is worthwhile to note that all the other combinations also do outperform CESI, which is the current state of the art model.

	Macro F1	Micro F1	Pair F1
CESI	0.627	0.844	0.819
CUVA	0.661	**0.867**	**0.855**
Glove+TransE+HAC	**0.662**	0.862	0.837
Glove+HolE+KMeans	0.633	0.855	0.826
FastText+HolE+HAC	0.651	0.858	0.823

Table 11: Results on the performance of CUVA while using other initialization strategies. The results are reported on the *Entity Canonicalization* task for *head entities only* on ReVerb45k.

Figure 5 illustrates the comparison of Macro F1 results for the Entity Canonicalization task on *all entity* mentions for the *test fold* of the Ambiguous dataset as a function of θ_E. Here, θ_E denotes the threshold cutoff used during HAC based initializations, which in turn sets the value for the number of mixtures K_E in CUVA. We donot highlight the Micro or Pair F1 values, since the relative change in those values while varying θ_E was minimal.

Its interesting to note that setting $\theta_E = 0.2$ yields a better Macro F1 value (by 1%) on the *test* fold, even though $\theta_E = 0.3$ had the best performance on the *validation* fold.

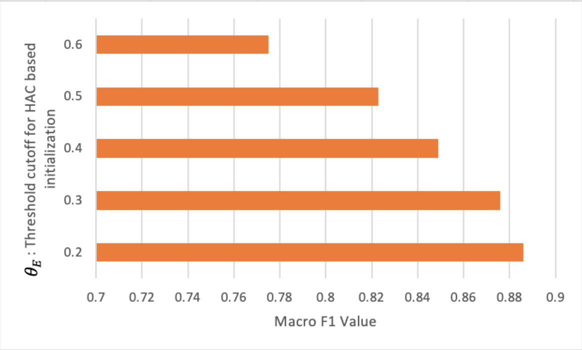

Figure 5: Comparison of Macro F1 results for the Entity Canonicalization task on *all entity* mentions for the Ambiguous dataset as a function θ_E, where θ_E denotes the threshold cutoff used during HAC based initialization and in turn sets the value for the number of entity clusters used in CUVA. All other hyperparams remain identical to the values denoted in Table 10. The list of Macro F1 values reported here have a standard deviation of 4.6%.

Furthermore, from Section A (of the Appendix), we note that CUVA uses several sources to generate additional side information, and utilizes it while training. Specifically, CUVA uses two *external* resources, i.e. an off the shelf Stanford CoreNLP entity linker (EL) (Spitkovsky and Chang, 2012) and a lexical resource called PPDB 2.0 (Pavlick et al., 2015), which is a collection of equivalent paraphrases. Table 12 illustrates the performance of CUVA when each of these *external* resources is ablated one at a time.

From these results, it is clear that the entity linker (EL) has a bigger impact on the results as opposed to PPDB 2.0 resource. This is because, being a statistical model, the Stanford CoreNLP entity linker

Approaches	Macro	Micro	Pair
CUVA	0.661	0.867	0.855
Without PPDB2	0.668	0.867	0.841
Without EL	0.595	0.845	0.846
Without EL and PPDB2	0.592	0.844	0.827

Table 12: Ablation tests demonstrating the effects of using external *resources* on the Entity Canonicalization task (*head mentions only*) for the ReVerb45K dataset.

is much more likely to capture lexical variations within mentions of the same entity, as opposed to the PPDB 2.0 lexical resource.