GATES: Using Graph Attention Networks for Entity Summarization

Asep Fajar Firmansyah
Data Science Research Group
Paderborn University
Germany
asep.fajar.firmansyah@uni-paderborn.de

Diego Moussallem
Data Science Research Group
Paderborn University
Germany
Globo
Rio de Janeiro, Brazil
diego.moussallem@uni-paderborn.de

Axel-Cyrille Ngonga Ngomo
Data Science Research Group
Paderborn University
Germany
axel.ngonga@upb.de

ABSTRACT

The sheer size of modern knowledge graphs has led to increased attention being paid to the entity summarization task. Given a knowledge graph T and an entity e found therein, solutions to entity summarization select a subset of the triples from T which summarize e's concise bound description. Presently, the best performing approaches rely on sequence-to-sequence models to generate entity summaries and use little to none of the structure information of T during the summarization process. We hypothesize that this structure information can be exploited to compute better summaries. To verify our hypothesis, we propose GATES, a new entity summarization approach that combines topological information and knowledge graph embeddings to encode triples. The topological information is encoded by means of a Graph Attention Network. Furthermore, ensemble learning is applied to boost the performance of triple scoring. We evaluate GATES on the DBpedia and LMDB datasets from ESBM (version 1.2), as well as on the FACES datasets. Our results show that GATES outperforms the state-of-the-art approaches on 4 of 6 configuration settings and reaches up to 0.574 F-measure. Pertaining to resulted summaries quality, GATES still underperforms the state of the arts as it obtains the highest score only on 1 of 6 configuration settings at 0.697 NDCG score. An open-source implementation of our approach and of the code necessary to rerun our experiments are available at https://github.com/dice-group/GATES.

CCS CONCEPTS

• **Information systems** → **Content ranking**; • **Computing methodologies** → *Semantic networks.*

KEYWORDS

Entity Summarization, Graph Attention Network, Knowledge Graph Embeddings, Text Embeddings.

K-CAP '21, December 2–3, 2021, Virtual Event, USA.

ACM ISBN 978-1-4503-8457-5/21/12…$15.00
https://doi.org/https://doi.org/10.1145/3460210.3493574

ACM Reference Format:
Asep Fajar Firmansyah, Diego Moussallem, and Axel-Cyrille Ngonga Ngomo. 2021. GATES: Using Graph Attention Networks for Entity Summarization. In *Proceedings of the 11th Knowledge Capture Conference (K-CAP '21), December 2–3, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/https://doi.org/10.1145/3460210.3493574

1 INTRODUCTION

Entity summarization (ES) approaches aim to generate size-constrained summaries of entities e described in an input knowledge graph T by selecting a subset of the assertions (i.e., subject-predicate-object triples) associated with e from T [18]. ES approaches are used in an increasing number of applications (e.g., the Google Knowledge Panels, RDF browsers such as Genesis [7]) to provide succinct representations of entities. Most recent works in this field rely on deep learning [15, 18, 30, 31]—especially on bidirectional Long Short-Term Memory (BiLSTM) networks [8]—to achieve general-purpose entity summarization. These recent works applying deep learning substantially outperform previous research based on unsupervised learning [5, 9, 10, 23, 29].

Still, the intuition behind the input representation from triples that represent an entity used in most current approaches goes back to MPSUM [29], which assumes the uniqueness of predicates and the importance of objects to be key factors while selecting the triples that are to be part of an entity's summary. For example, the approaches presented in [30] and [31] encode the predicates and objects of triples using a combination of text embeddings and knowledge graph embeddings (e.g., word2vec [3] and TransE [28], respectively). The authors of DeepLENS [18] and [15] exploit the idea of distributional semantics by encoding triples with fastText [12].

The *core observation* behind this paper is that none of the current approaches exploits adjacency information in T explicitly. For example, the premise behind the state-of-the-art approach DeepLENS is that textual information is "more important than graph structure" [18, p. 2]. While the occurrence of entities across triples provides some local information on adjacency, global information on the structure of T has not been exploited for entity summarization. We hypothesize that *including more structure information into the entity summarization process can lead to better entity summaries* by providing hints pertaining to the amount of information in a given triple at the scale of T.

To validate our hypothesis, we developed GATES. Our approach is inspired by [33], which uses Graph Convolutional Networks (GCN) [13] to generate multi-document summarizations. The authors exploit deep neural networks to encode the sentences in documents and combine them with a graph representation of document clusters to produce multi-document summaries. GATES applies Graph Attention Networks (GAT) [26] to rank triples while taking the structure of the input graph T into consideration. To our knowledge, we are the first to implement deep learning for graphs on the ES task. In addition to leveraging the adjacency matrix of T to collect structure information, we combine knowledge graph embeddings and text embeddings to represent triples. By these means, we capture the information in triples (like previous works) and benefit from the topology of T. In contrast to recent works [18, 30], GATES applies knowledge graph embeddings and text embeddings on triple encoding to encode triples' predicates and objects, respectively.

We evaluated GATES on the DBpedia and LMDB datasets from ESBM (version 1.2) [16] as well as the FACES datasets [9]. Our results suggest that we outperform the current state of the art on both datasets and reach up to 0.574 F-measure. The main contributions of this paper can be summarized as follows:

- We present a new approach for entity summarization dubbed GATES. Our approach is able to exploit the structure of knowledge graphs via a graph attention network.
- GATES combines graph structure information with knowledge graph embeddings and text embeddings to encode triples quality of entity summarization.
- Our ablation study suggests that the use of weighted adjacency matrix, the combination of ComplEx graph embedding and GloVe text embedding in triple encoding, and single-head attention in GAT are central for GATES outperforming the state of the art.

2 RELATED WORK

The research on entity summarization was dominated by approaches based on unsupervised learning up to and including 2016. RELIN [5] applies a centrality-based method based on the random surfer model [20] to compute relatedness and informativeness features that are used to compute a triple ranking. FACES [9] combines three dimensions: diversity, uniqueness, and popularity to produce the summaries. It uses the incremental hierarchical conceptual clustering algorithm [19] to identify these three dimensions. A lightweight link-based approach for the relevance-oriented summarization of knowledge graph entities is described by LINKSUM [24]. The approach MPSUM [29] utilizes the Latent Dirichlet allocation (LDA) model for entity summarization and proposed MP (match up objects and RDF triples on predicates) that considers predicate uniqueness and object importance in RDF triple to generate triple ranking.

More recent approaches generate summaries within a supervised setting. At the time of writing, published approaches have solely used BiLSTM networks as the core of their entity summarizer models. To the best of our knowledge, ESA [30] is the first model that implements a neural network method in this research area leveraging a BiLSTM network. Inspired by MPSUM [29], the authors used predicates and objects of RDF triples (facts) as key factors to consider when computing summaries. By combining text embeddings [3] and knowledge graph embeddings [4], they encoded entity descriptions consisting of predicate-object pairs for each triple in the input entity's concise bound description into vectors as embeddings. Afterward, the embeddings are used as inputs to the BiLSTM network with an attention mechanism to calculate triple scoring. ESA outperforms unsupervised methods that are evaluated on ESBM (version 1.1)[1] [16]. Inspired by ESA, DeepLENS [18] also leverages a BiLSTM to encode the entity description and leverages multilayer perceptron (MLP) to compute triple scoring. Unlike ESA, DeepLENS considers applying textual semantics rather than using knowledge graph embeddings to carry the entity summarization tasks. DeepLENS implements fastText [12] as triple encoding to capture information meaning in entity descriptions and feed it into the BiLSTM network. The authors evaluated DeepLENS using ESBM (version 1.2)[2] [16]. Their model outperforms ESA on both datasets from ESBM, i.e., DBpedia and LinkedMDB (LMDB). AutoSUM [31] extends ESA and consists of two modules: an extractor and a simulator. The first module extracts features from input, which are represented in triples, using a BiLSTM. The simulator takes the extracted features to simulate multi-user preferences. AutoSUM is able to achieve ESA on all datasets on ESBM (version 1.1). Similarly, DRESSED, an extension of DeepLENS, leverages user feedback as input to perform triple scoring. Moreover, to calculate the triple scoring, it applies deep reinforcement learning to combine results from entity summarizer and user actions. DRESSED is evaluated on modified ESBM (version 1.2) datasets and FED dataset [15].

3 THE GATES APPROACH

3.1 Overview

3.1.1 Definitions. Let E be a set of entities, R be a set of relations, and C and L be sets of classes and literals, respectively. We call $T \subseteq E \times R \times (C \cup L \cup E)$ a knowledge graph. Given the triple $(s, p, o) \in T$, we call s the subject, p the predicate and o the object of the triple. We define the set $V(T)$ of entities found in T as follows: $V(T) = \{e \in E : \exists e' \in E \; \exists r \in R \text{ with } (e, r, e') \in T \vee (e', r, e) \in T\}$. We write $Desc(e, T)$ to mean the concise bound description of the entity e in T. Note that $Desc(e, T) \subseteq T$. For the sake of brevity, we write $Desc(e)$ instead of $Desc(e, T)$ when T is unambiguous.

3.1.2 Problem Statement. Given a knowledge graph T, a size constraint $k \in \mathbb{N}^+$ and an entity $e \in V(T)$, the goal of GATES is to generate an optimal summary $ES(e) \subseteq Desc(e)$ with $|ES(e)| \leq k$. To compute $ES(e)$, GATES transforms T into a graph representation and employs a triple encoding mechanism (detailed in Section 3.4.2). Subsequently, it uses the triple encoding and the graph structure of T in a GAT to compute high-level features of triples as well as to rank triples. $ES(e)$ are the top-k triples based on the output of the GAT.

3.2 Background

3.2.1 Graph Attention Network. We apply a Graph Attention Network (GAT) [26] to generate high-level features for triple scoring. In this subsection, we explain the implementation of the GAT in

[1] https://github.com/nju-websoft/ESBM/tree/master/v1.1
[2] https://github.com/nju-websoft/ESBM/tree/master/v1.2

our model, and how this GAT produces scores for the triples in the description of entities. Like Graph Convolutional Networks (GCN) [13, 33], GATs require two inputs: an adjacency matrix and node features. An entry a_{ij} of the adjacency matrix $A \in \mathbb{R}^{N \times N}$ of a graph $\mathcal{G}$ (where N stands for the number of nodes in $\mathcal{G}$) describes how often the node v_i of $\mathcal{G}$ is the source of an edge whose end is v_j.

We denote the feature vector of the node v_i by $h_i \in \mathbb{R}^F$ where F is the number of features used to encode a node. The input layer for a GAT is hence $h = \{h_1, h_2, ..., h_N\}$. The GAT computes a set h' of new node features as its output $h^{'} = \{h_1^{'}, h_2^{'}, ..., h_N^{'}\}$, where $h_i^{'} \in \mathbb{R}^{F^{'}}$ and $F^{'} \in \mathbb{N}$ is a number of new node features. Initially, a shared linear transformation, parameterized by a weight matrix $W \in \mathbb{R}^{F^{'} \times F}$ is applied to every node. We then perform self-attention on the nodes while relying on a shared attention mechanism $a : \mathbb{R}^{F^{'}} \times \mathbb{R}^{F^{'}} \rightarrow \mathbb{R}$. We set

$$e_{ij} = a(Wh_i, Wh_j), \tag{1}$$

where $a(.,.)$ refers to the attention mechanism and e_{ij} is the calculated attention coefficient. Node's attention coefficient and its first-hop neighbors are computed in order to preserve topological information of graphs [14]. A softmax function is applied to normalize the attention coefficients by converting them into probabilities to make them comparable. Another function, called Leaky ReLU activation, is implemented to acquire the final normalized attention coefficient α_{ij}.

$$\alpha_{ij} = softmax(LeakyReLU(e_{ij})) \tag{2}$$

Afterward, the normalized attention coefficients are used to compute a linear combination of node features. We also apply multi-head attention on implementing GAT in our model to discover its contribution. Therefore, we calculate the set of new features, $h^{'}$, as given by

$$h_i^{'} = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j \in neighbors(i)} \alpha_{ij}^k W^k h_j\right), \tag{3}$$

where $neighbors(i)$ stands for the set of nodes which are adjacent to the node v_i and $\|$ denotes the concatenation of vectors. In addition, a non-linearity σ is applied to the summation results.

3.3 Architecture

In this section, we elucidate the formal workings of GATES. Fig. 1 gives an overview of GATES' architecture, which has a modular design comprising

(1) the generation of the input representation, including the computation of the adjacency matrix generation of the encoding of triples,
(2) the computation of scores for triples and
(3) the generation of entity summaries

We discuss each component in detail in the following subsections.

3.4 Input representation

3.4.1 Adjacency Matrix. We compute T's adjacency matrix as follows: Let $V(T)$ be the set of entities contained in T, ergo

$$V(T) = \{v : \exists(v, p, o) \in T \vee \exists(s, p, v) \in T\}. \tag{4}$$

We define the entries a_{ij} of T's adjacency matrix $A(T)$ as follows:

$$a_{ij} = \begin{cases} 1 \text{ if } (v_i, p, v_j) \in T \vee (v_j, p, v_i) \in T; \\ 0 \text{ otherwise.} \end{cases} \tag{5}$$

In addition to a binary adjacency matrix, we also consider a weighted version of the adjacency matrix [1] and evaluate whether it is beneficial for GATES in a series of experiments reported in Section 4. The weight of a property p is computed using its $tfidf$ score relative to a node $v \in V(T)$. The term frequency $tf(p, u, T)$ encapsulates the number of times that u is the subject of a triple with the property p in T. The inverse document frequency of p describes how important said property is in T. Formally, the term frequency $tf(p, u, T)$ is defined by:

$$tf(p, u, T) = \frac{|\{o : (u, p, o) \in T\}|}{|\{(o, q) : (u, q, o) \in T\}|}. \tag{6}$$

The inverse document frequency of p is given by

$$idf(p, T) = \ln \frac{|V(T)|}{|\{v \in V(T) : (v, p, o) \in T\}|}. \tag{7}$$

We now define $tfidf(p, u, T)$ as

$$tfidf(p, u, T) = tf(p, u, T) \times idf(p, T). \tag{8}$$

The entries of the weighted adjacency matrix a_{ij} are now given by

$$a_{ij} = \begin{cases} \sum\limits_{p:(v_i,p,v_j) \in T} tfidf(p) \text{ if } \exists p : (v_i, p, v_j) \in T, \\ 0 \text{ otherwise.} \end{cases} \tag{9}$$

To work with weighted adjacency matrix, we follow [27] where we utilize the weighted edges value as edge features to update node features in the attention-mechanism process. Let edge features D be a set of non-zero value d_{ij} obtained from the weighted adjacency matrix, where $d_{ij} \in \mathbb{R}^{F_E}$ and $F_E \in \mathbb{N}$ is the number of edge features. By using linear transformation, D is transformed to higher-level features. In doing so, we apply weight matrices $W_e \in \mathbb{R}^{F_E \times F_E}$ to every edge. Both edge and node features are used in the attention mechanism $a : \mathbb{R}^{F^{'}} \times \mathbb{R}^{F^{'}} \times \mathbb{R}^{F_E^{'}} \rightarrow \mathbb{R}$ to generate the attention coefficient. This process is performed on each node as follows:

$$e_{ij} = a(Wh_i, Wh_j, W_e d_{ij}), \tag{10}$$

We normalize the attention coefficient using equation 2. Finally, we calculate the set of new features, h , as given by

$$h_i^{'} = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j \in neighbors(i)} \alpha_{ij}^k W^k h_j W_e^k d_{ij}\right), \tag{11}$$

3.4.2 Triple encoding. We exploit both knowledge graph and text embeddings to encode the triples in $Desc(e)$ into continuous vectors. Let t_i be in a triple in $Desc(e)$. Moreover, let $GE : R \rightarrow \mathbb{C}^{k_1}$ be a knowledge graph embedding function able to represent properties explicitly. We represent k_1-dimensional complex vectors as $2k_1$-dimensional real vectors by concatenating the k_1 real and k_1 imaginary components of the complex vectors. Moreover, let $TE : E \rightarrow \mathbb{R}^{k_2}$ be a text embedding function able to embed entities. The embedding of $t_i = (e, p_i, o_i) \in Desc(e)$ is set to

$$GE(p_i)||TE(o_i) \tag{12}$$

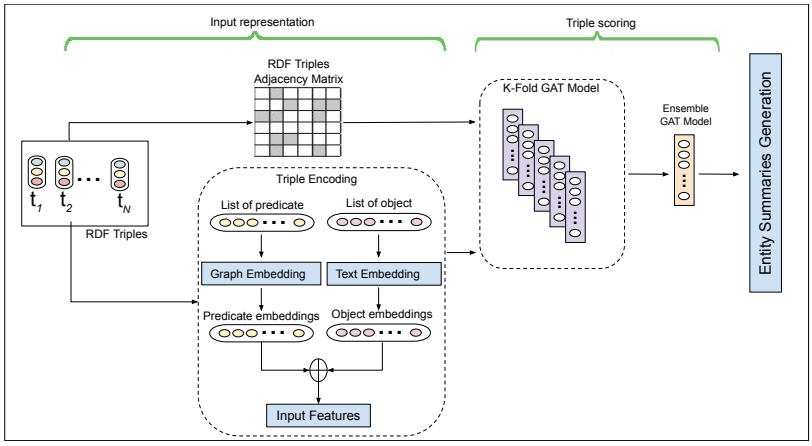


Figure 1: Architecture of entity summarization using GAT (GATES).

.

and is of dimension $2k_1+k_2$. While computing embeddings using the approaches above is straightforward, we employ the same method as [18] to define of the textual form of an object value. If the object value is identified by Internationalized Resource Identifiers (IRIs), we retrieve one of its `rdfs:label`. However, if the object value does not have a label, we set its local name as the textual form. For objects that exist in the form of literal, we take their lexical form. Subsequently, each word of the textual form is mapped into vectors using a pre-trained word embedding model before averaging the vectors to obtain one vector representation.

In our implementation, we employ three knowledge graph embeddings: DistMult [32], ComplEx [25], and ConEx [6]). DistMult is a graph embedding that uses neural embedding models and focuses on symmetric relations. Complex embedding effectively manages complex number using the Hermitian dot product to handle symmetric, reflexity, and irreflexity relations. On the other hand, it deals also with antisymmetric relations. Furthermore, ConEX is a complex-valued convolutional neural model that learns complex-valued vector representation of a given knowledge graph by combining a 2D convolution operation with a Hermitian dot product [6].

We employ two text embeddings: GloVe and fastText. GloVe [21] combines global matrix factorization and local context window methods to produce word vector space with meaningful substructure. In our work, we use a pre-trained word vectors glove.6B model that is trained using Wikipedia 2014 and Gigaword 5. fastText [12] is a skip-gram-based word representations technique where each word is represented as a bag of character n-grams. This method aims to capture the meaning of short words. In our work, we use a pre-trained fasText model that contains 1 million word vectors trained on Wikipedia 2017.

3.5 Triple scoring

We learn the scoring function for triples in a manner akin to [30]. First, the output of the GAT for each triple is normalized using the softmax function. Therewith, we compute a distribution of the probability of triples belonging to the summary we aimed to compute. We then approximate to generate gold summaries by using the cross-entropy loss against the gold standard probabilities of triples belonging to the summary of each of the entities in the gold standard.

In order to increase the accuracy of triple scoring, we implement a five-fold cross-validation methodology. Five separate GATES models (k-5) are resulted in accordance to each fold of training data. Instead of applying a single model-based learner, we follow the approach from [2]. Thus, we compute the prediction by involving all five models. In doing so, we use a soft voting ensemble technique that has been proven to improve prediction accuracy over a single model alone.

3.6 Generation of Entity summaries

We can finally define the summary $ES(E)$ of an entity e based on $Desc(e)$ in a manner akin to [17] by selecting the top-k $Desc(e)$ triples w.r.t. the score computed in the section above. $ES(e)$ is hence a k-subset of $Desc(e)$ with

$$\forall t' \in ES(e) \backslash Desc(e) : score(t') \leq \min_{t \in Desc(e)} score(t). \qquad (13)$$

4 EVALUATION

4.1 Goal

The goal of our evaluation was to ascertain whether graph structure information can improve the quality of entity summaries. To achieve this goal, we compared GATES with the supervised state-of-the-art techniques, ESA [30], AutoSUM[31], and DeepLENS [18][3]. The benchmark provides both training and evaluation datasets for two knowledge graphs, i.e., DBpedia and LinkedMDB (LMDB).

4.2 Datasets and Evaluation Metrics

In this study, we used ESBM (version 1.2) [16] and FACES [9] as benchmarks. ESBM contains 6,584 triples distributed across 175 entities. 150 entities are from DBpedia and 25 entities are from LMDB. On the other hand, FACES has 50 entities and 2,152 triples. Both datasets are designed to support model evaluation using five-fold

[3]Note that we do not compare with DRESSED because it assumes an interactive setting.

cross-validation and provide two kinds of ground-truth summaries: summaries of length 5 and 10. For each entity e, ESBM provides six manually generated summaries, while FACES contains four to eight manually generated summaries. To ensure that our experimental results are reproducible, we did not modify the ESBM benchmark in any way and used the evaluation function provided by the ESBM benchmark[4] to compute F-measures and NDCG (Normalized Discounted Cumulative Gain) scores throughout our experiments. Since the evaluation function of the benchmark does not serve FACES's evaluation, we applied the same algorithm to perform the evaluation.

4.3 Experimental Setup

This section describes the experimental settings that were used to train GATES. We trained on the complete knowledge graph contained in the benchmark to compute knowledge graph embeddings models using ComplEx [25], DistMult [32], and ConEx [6] with the embedding size set to 300. We used this setting because it has been used in the literature of knowledge graph embeddings [6]. For text embeddings, we used 300 dimensions for both GloVe and fastText based on previous works [18].

We conducted a hyperparameter optimization to discover default experimental settings for GATES. We optimized for single and multi-head attention, the weights of the adjacency matrix, the number of hidden layers, the learning rate, and the use of regularization. Details are given in section 4.4.1. Furthermore, we used the Adam optimizer and binary cross-entropy (BCE) as loss function. To avoid overfitting without lowering the number of learning instances[11, 31], GATES was trained using the five-fold cross-validation technique. Moreover, the early stopping method was applied on validation sets during the training. The final hyperparameters that we used in our experiments were a learning rate of 0.05, 2 hidden layers, single-head attention, a dropout rate of 0 and no regularization.

To evaluate whether the results achieved by GATES differ significantly from those of the state-the-art approaches, we performed a statistical significance test. We used the Wilcoxon signed ranked test [22] with a significance threshold of 95%. All experiments were carried on a 8-core Intel i5 8250U CPU (1.60 GHz) with 15.5 GB of RAM running on Ubuntu 20.04.2 LTS (64-bit) without CUDA.

4.4 Experimental Results

We begin by presenting some of most important results while performing the hyperparameter configuration of GATES. Thereafter, we compare GATES with ESA, AutoSUM, and DeepLENS.

4.4.1 Hyperparameters. We used six different triple encoding settings by combining each of the three knowledge graph embeddings and two text embeddings we considered in this work. In a first series of experiments, we used the binary version of the adjacency matrix. We fixed the learning rate to 0.05 and we compared all GATES models with both single-head and multi-head attention at the following hyperparameter ranges: number of hidden layers $\in \{2, 4\}$, L2-regularization $\in \{0, 1 \times 10^{-5}\}$, and number of attention heads $\in \{1, 2, 4, 6\}$.

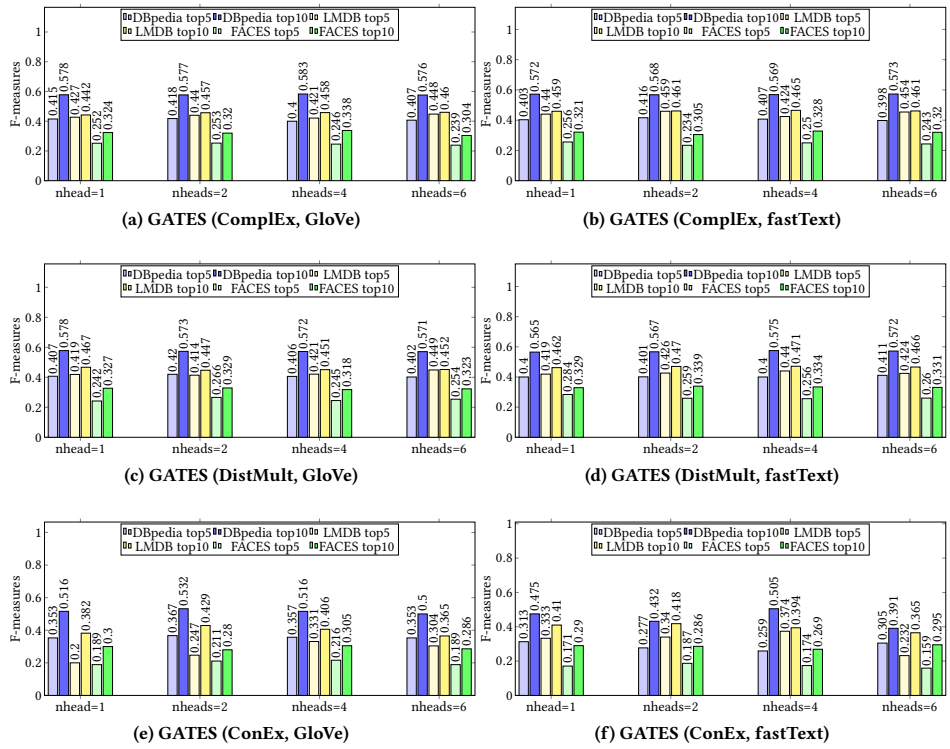
[4]https://github.com/nju-websoft/ESBM/tree/master/v1.2/Evaluator

Fig. 2 gives an overview of our findings for this first series of experiments. Using GloVe led the model to have slightly higher F-measures than when it used fastText across different settings for the number of attention heads. On average, the effective configurations to improving the F-measures were by using 1, 2, and 4 heads in the multi-head attention layer. In a second set of experiments, we used weighted adjacency matrices where the edge weights were computed using $tfidf$ (Equation 8). We then compared the different settings of GATES with the same hyperparameters as above. The results shown in Fig. 3 display the performance of GATES for different settings. On average, GATES achieves better F-measures with the weighted version of the adjacency matrix. The ranking across the combinations of knowledge graph and text embeddings are in line with those achieved in the first experiments.

To select the setting for GATES in all further experiments, we selected the highest F-measure achieved by GATES using all six combination of embeddings in Table 1. The combination of ComplEx and GloVe embeddings achieves the best average rank across all experiments by ranking first in 4 of 12 experiments, second in five experiments, third in two experiments, and fourth in one experiment. It is also the best configuration on top-5 summaries of DBpedia with binary and weighted adjacency matrix, and also on the top-5 of LMDB with binary adjacency matrix, and the top-10 of LMDB with weighted adjacency matrix. DistMult and fastText embeddings are together able to achieve the best performance on FACES dataset with a weighted adjacency matrix and the top-10 of LMDB with a binary adjacency matrix. The combination of DistMult and GloVe embeddings outperforms all other models on all top-k summaries of DBpedia with a binary adjacency matrix, and also in the top-10 summaries of DBpedia with a weighted adjacency matrix, and for top-5 summaries of FACES with a binary adjacency matrix. In addition, the combination of ComplEx and fastText has achieved all other models on all top-k summaries of LMDB with a binary adjacency matrix, and also on top-10 summaries of FACES. Still, the verdict is clear: GATES performs best with a weighted adjacency matrix, ComplEx as graph embedding and GloVe as text embedding. We hence use this model with 2 hidden layers, a single-head attention, a dropout rate of 0, and no regularization in our subsequent experiments.

Table 2 presents NDCG scores that are used to measure the quality of obtained summaries ranking. In a comparison with the binary adjacency matrix, applying a weighted adjacency matrix on GATES is proven to slightly increase the summaries ranking quality. Furthermore, the combination of ComplEx and GloVe embeddings outperforms other models where it achieves the highest average score of NDCG in 7 of 12 experiments. DistMult and fastText embeddings combination is in the second place where it achieves the best average score in 4 of 12 experiments. Meanwhile, the implementation of ComplEx and fastText gains the best average score only in two experiments. The last combination, DistMult and GloVe embeddings, obtains the best average score only in one experiment.

4.4.2 Comparison with state-of-the-art approaches. We compared GATES with the following state-of-the-art approaches: ESA, AutoSUM, and DeepLENS. All approaches use supervised training methods. Since there are no NCDG scores available for the three approaches, we re-ran the experiments of all approaches with



(a) GATES (ComplEx, GloVe) (b) GATES (ComplEx, fastText)

(c) GATES (DistMult, GloVe) (d) GATES (DistMult, fastText)

(e) GATES (ConEx, GloVe) (f) GATES (ConEx, fastText)

Figure 2: F-measures achieved by all configurations of GATES in the first series of experiments

Table 1: Highest F-measure of all combinations of knowledge graph and text embeddings in our first (binary adjacency matrix) and second (weighted adjacency matrix) series of experiments.

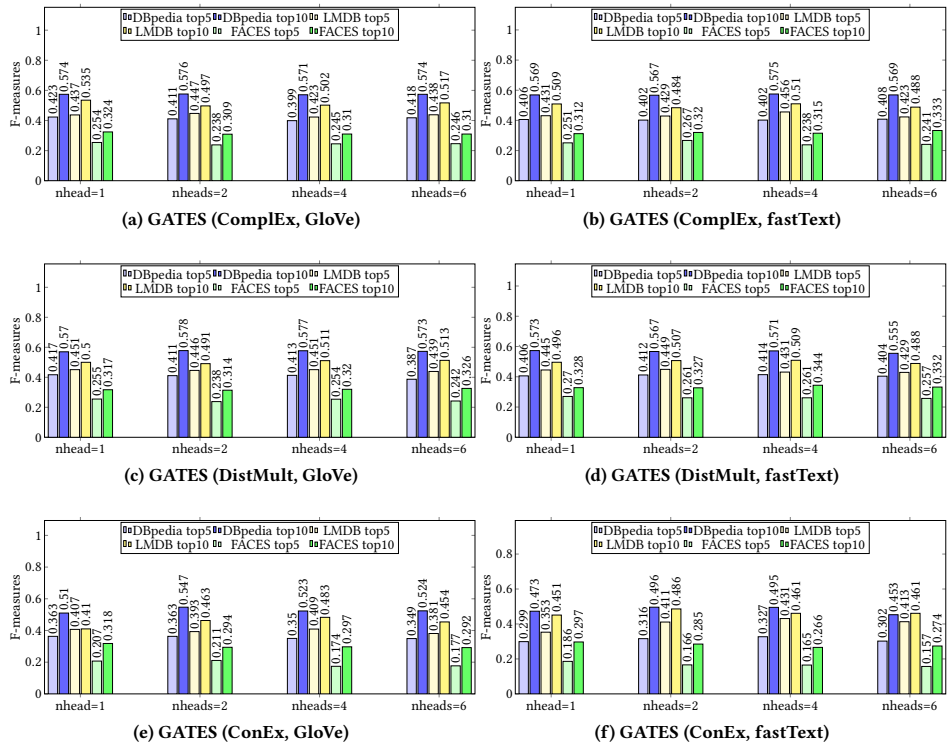
	DBpedia		LMDB		FACES	
	k=5	k=10	k=5	k=10	k=5	k=10
Binary adjacency matrix						
GATES (ComplEx, GloVe)	**0.418**	0.577	**0.440**	0.457	0.253	0.320
GATES (DistMult, GloVe)	**0.418**	**0.578**	0.435	0.449	**0.263**	0.317
GATES (ConEx, GloVe)	0.357	0.516	0.331	0.406	0.216	0.305
GATES (ComplEx, fastText)	0.407	0.569	**0.440**	**0.466**	0.258	**0.333**
GATES (DistMult, fastText)	0.411	0.572	0.424	**0.466**	0.260	0.331
GATES (ConEx, fastText)	0.313	0.475	0.333	0.410	0.171	0.290
Weighted adjacency matrix						
GATES (ComplEx, GloVe)	**0.423**	0.574	0.437	**0.535**	0.254	0.324
GATES (DistMult, GloVe)	0.413	**0.577**	**0.451**	0.511	0.254	0.310
GATES (ConEx, GloVe)	0.361	0.525	0.393	0.489	0.220	0.300
GATES (ComplEx, fastText)	0.405	0.576	0.432	0.510	0.253	0.318
GATES (DistMult, fastText)	0.414	0.571	0.431	0.509	**0.261**	**0.344**
GATES (ConEx, fastText)	0.326	0.499	0.370	0.443	0.173	0.284

Table 2: Highest NDCG scores of all combinations of knowledge graph and text embeddings in our first (binary adjacency matrix) and second (weighted adjacency matrix) series of experiments.

	DBpedia		LMDB		FACES	
	k=5	k=10	k=5	k=10	k=5	k=10
Binary adjacency matrix						
GATES (ComplEx, GloVe)	**0.803**	**0.893**	**0.818**	**0.869**	0.693	0.754
GATES (DistMult, GloVe)	0.794	0.892	0.799	0.856	0.703	0.760
GATES (ConEx, GloVe)	0.738	0.845	0.701	0.801	0.652	0.747
GATES (ComplEx, fastText)	0.795	0.889	0.801	**0.869**	0.694	0.756
GATES (DistMult, fastText)	0.787	0.886	0.806	0.867	**0.708**	**0.766**
GATES (ConEx, fastText)	0.702	0.809	0.727	0.821	0.613	0.739
Weighted adjacency matrix						
GATES (ComplEx, GloVe)	**0.798**	0.893	0.804	**0.881**	**0.697**	0.759
GATES (DistMult, GloVe)	0.782	0.890	**0.818**	0.878	0.702	0.759
GATES (ConEx, GloVe)	0.737	0.845	0.744	0.832	0.645	0.739
GATES (ComplEx, fastText)	0.791	**0.900**	0.800	0.878	0.686	0.757
GATES (DistMult, fastText)	0.786	0.885	0.814	0.877	**0.697**	**0.769**
GATES (ConEx, fastText)	0.737	0.843	0.737	0.825	0.617	0.731

the according configuration settings from the respective papers [30][31][18]. Subsequently, we evaluated the three approaches with ESBM (version 1.2) and FACES datasets. As shown in Table 3, GATES outperforms all approaches on top-5 summaries of DBpedia, top-10 summaries of LMDB, and all top-k summaries of FACES. The improvement of GATES over DeepLENS was particular large for $k = 5$ on DBpedia. We performed a Wilcoxon signed rank test to check whether our results were significant. The null hypothesis of the test was that the F-measures achieved by GATES on each of the summaries came from the same distribution as those achieved by ESA, AutoSUM, or DeepLENS. We were able to negate this null hypothesis ($p \leq 0.05$) on top-10 summaries of LMDB and FACES, and on top-5 summaries of all datasets for ESA. Moreover, GATES significantly outperforms AutoSUM on top-10 summaries of DBpedia (see Table 3).

Table 4 presents NDCG results on the benchmark for comparing the approaches. Although GATES is inferior to DeepLENS in most settings, GATES outperforms DeepLENS on FACES dataset.



(a) GATES (ComplEx, GloVe)

(b) GATES (ComplEx, fastText)

(c) GATES (DistMult, GloVe)

(d) GATES (DistMult, fastText)

(e) GATES (ConEx, GloVe)

(f) GATES (ConEx, fastText)

Figure 3: F-measures achieved by all six configurations of GATES in the second set of experiments

Table 3: Average F-measure score based on our model testing via five-fold cross-validation processes to all entities of the benchmark. Statistically significant results ($p < 0.05$) are indicated by ▲. Results that are not significant are marked with ○. Underperform results are denoted with ▼. The first, second and third symbols next to a result in the last row encode the statistical significance of the comparison of GATES with ESA, AutoSUM and DeepLENS, respectively.

	DBpedia		LMDB		FACES	
	k-5	k=10	k=5	k=10	k=5	k=10
F-Measure						
ESA	0.332	0.532	0.353	0.435	0.153	0.261
AutoSUM	0.372	0.555	0.430	0.520	0.241	0.316
DeepLENS	0.404	**0.575**	**0.469**	0.489	0.130	0.248
GATES	**0.423**	0.574	0.437	**0.535**	**0.254**	**0.324**
	▲ ○ ○	▲▲▼	▲ ○ ▼	▲▲▲	▲ ○ ▲	▲▲▲

Table 4: Average NDCG score based on our model testing via five-fold cross-validation processes to all entities of the benchmark.

	DBpedia		LMDB		FACES	
	k-5	k=10	k=5	k=10	k=5	k=10
F-Measure						
ESA	0.755	0.846	0.737	0.799	0.601	0.707
AutoSUM	0.797	0.882	0.809	0.856	0.693	**0.768**
DeepLENS	**0.825**	**0.905**	**0.855**	**0.888**	0.585	0.715
GATES	0.798	0.893	0.804	0.881	**0.697** (▲0.58%)	0.759

4.5 Complexity Analysis

We performed time complexity analysis of GATES. Given a knowledge graph T, we can compute $A(T)$ in $O(|V(T)|^2)$ and need $O(|T|)$ space to store it. Computing the weighted version of $A(T)$ has the same worst-case time and space complexity. We assume that the embeddings are pre-computed and can fetch them in constant time for each element of $R \cup E$, ergo need $O(|T|)$ in space and time to collect the embeddings which serve as input for out GAT. Each iteration of the computation in the GAT is carried out in $O(nheads \times F^{'} F|V(T)|^2)$, where $nheads$ is the number of attention heads. We also measured the runtime of GATES in our experiments. The results shown in Table 5 suggest that our approach can be trained in under 10 min in all experiments.

Table 5: Runtime Analysis of GATES on the benchmark. All times are seconds.

Dataset	Input Triples	Output Triples	Training Time (in second)		Epochs
			Total	Mean	
DBpedia top-5	4436	750	657.96	2.631	250
DBpedia top-10	4436	1500	682.98	2.732	250
LMDB top-5	2148	125	354.87	1.419	250
LMDB top-10	2148	250	362.45	1.450	250
FACES top-5	2152	125	361.71	1.447	250
FACES top-10	2152	250	403.36	1,613	250

5 CONCLUSION AND FUTURE WORKS

We presented GATES, a new approach on general-purpose entity summarization task based on extractive method. Our approach introduces a triple encoding mechanism that combines knowledge graph embeddings and text embeddings, and implements deep learning on graphs using a GAT. Based on our ablation study, we discovered the model achieves the best performance when it was equipped with 1) the adjacency matrix with weighted edges, 2) the combination of ComplEx as knowledge graph embedding and GloVe as text embedding, 3) the use of single-head attention. GATES outperformed all of the state-of-the-art approaches on top-10 summaries of LMDB and FACES. In addition, GATES achieved ESA and AutoSUM on top-5 summaries of all datasets and on top-10 summaries of DBpedia, respectively. GATES reaches the highest F-measure of 0.574. In the future, we expect to implement GATES on bigger benchmarks and domain-specific datasets such as BioASQ[5]. As simple graphs have been proven able to improve entity summarization performance, we hence intend to extend that portion of our approach with hypergraphs [34].

ACKNOWLEDGEMENT

We acknowledge the support of the Federal Ministry for Economic Affairs and Energy (BMWi) project SPEAKER (FKZ 01MK20011A).

REFERENCES

[1] Mehmet Aydar, Serkan Ayvaz, and Austin Melton. 2015. Automatic Weight Generation and Class Predicate Stability in RDF Summary Graphs. In *Proceedings of the 4th International Workshop on Intelligent Exploration of Semantic Data (IESD 2015) co-located with the 14th International Semantic Web Conference (ISWC 2015), Bethlehem, Pennsylvania , USA, October 12, 2015 (CEUR Workshop Proceedings)*, Vol. 1472. CEUR-WS.org.

[2] Claudia Beleites and Reiner Salzer. 2008. Assessing and improving the stability of chemometric models in small sample size situations. *Analytical and bioanalytical chemistry* 390, 5 (2008), 1261–1271.

[3] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A Neural Probabilistic Language Model. *J. Mach. Learn. Res.* 3 (2003), 1137–1155. http://jmlr.org/papers/v3/bengio03a.html

[4] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Neural Information Processing Systems (NIPS)*. 1–9.

[5] Gong Cheng, Thanh Tran, and Yuzhong Qu. 2011. Relin: relatedness and informativeness-based centrality for entity summarization. In *International Semantic Web Conference*. Springer, 114–129.

[6] Caglar Demir and Axel-Cyrille Ngonga Ngomo. 2021. Convolutional Complex Knowledge Graph Embeddings. In *Eighteenth Extended Semantic Web Conference - Research Track*. https://openreview.net/forum?id=6T45-4TFqaX

[7] Timofey Ermilov, Diego Moussallem, Ricardo Usbeck, and Axel-Cyrille Ngonga Ngomo. 2017. GENESIS: a generic RDF data access interface. In *Proceedings of the International Conference on Web Intelligence*. 125–131.

[8] Alex Graves. 2013. Generating Sequences With Recurrent Neural Networks. *CoRR* abs/1308.0850 (2013). http://arxiv.org/abs/1308.0850

[9] Kalpa Gunaratna, Krishnaprasad Thirunarayan, and Amit Sheth. 2015. FACES: diversity-aware entity summarization using incremental hierarchical conceptual clustering. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. AAAI Press, 116–122.

[10] Kalpa Gunaratna, Krishnaprasad Thirunarayan, Amit P. Sheth, and Gong Cheng. 2016. Gleaning Types for Literals in RDF Triples with Application to Entity Summarization. In *The Semantic Web. Latest Advances and New Domains - 13th International Conference, ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Proceedings (Lecture Notes in Computer Science)*, Vol. 9678. Springer, 85–100. https://doi.org/10.1007/978-3-319-34129-3_6

[11] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A Deep Relevance Matching Model for Ad-hoc Retrieval. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*, Snehasis Mukhopadhyay, ChengXiang Zhai, Elisa Bertino, Fabio Crestani, Javed Mostafa, Jie Tang, Luo Si, Xiaofang Zhou, Yi Chang, Yunyao Li, and Parikshit Sondhi (Eds.). ACM, 55–64.

[12] Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hervé Jégou, and Tomás Mikolov. 2016. FastText.zip: Compressing text classification models. *CoRR* abs/1612.03651 (2016). http://arxiv.org/abs/1612.03651

[13] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=SJU4ayYgl

[14] Xiangyuan Kong, Weiwei Xing, Xiang Wei, Peng Bao, Jian Zhang, and Wei Lu. 2020. STGAT: Spatial-Temporal Graph Attention Networks for Traffic Flow Forecasting. *IEEE Access* 8 (2020), 134363–134372.

[15] Qingxia Liu, Yue Chen, Gong Cheng, Evgeny Kharlamov, Junyou Li, and Yuzhong Qu. 2020. Entity Summarization with User Feedback. In *The Semantic Web - 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings (Lecture Notes in Computer Science)*, Vol. 12123. Springer, 376–392.

[16] Qingxia Liu, Gong Cheng, Kalpa Gunaratna, and Yuzhong Qu. 2020. ESBM: an entity summarization benchmark. In *European Semantic Web Conference*. Springer, 548–564.

[17] Qingxia Liu, Gong Cheng, Kalpa Gunaratna, and Yuzhong Qu. 2021. Entity summarization: State of the art and future challenges. *J. Web Semant.* 69 (2021), 100647. https://doi.org/10.1016/j.websem.2021.100647

[18] Qingxia Liu, Gong Cheng, and Yuzhong Qu. 2020. DeepLENS: Deep Learning for Entity Summarization. In *Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG2020) co-located with the 17th Extended Semantic Web Conference 2020 (ESWC 2020), Heraklion, Greece, June 02, 2020 - moved online (CEUR Workshop Proceedings)*, Vol. 2635. CEUR-WS.org.

[19] Yoëlle S Maarek. 1990. An incremental conceptual clustering algorithm that reduces input-ordering bias. In *Advances in Artificial Intelligence*. Springer, 129–144.

[20] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web.* Technical Report. Stanford InfoLab.

[21] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1532–1543. https://doi.org/10.3115/v1/D14-1162

[22] Bernard Rosner, Robert J Glynn, and Mei-Ling T Lee. 2006. The Wilcoxon signed rank test for paired comparisons of clustered data. *Biometrics* 62, 1 (2006), 185–192.

[23] Marcin Sydow, Mariusz Pikuła, and Ralf Schenkel. 2010. DIVERSUM: Towards diversified summarisation of entities in knowledge graphs. In *2010 IEEE 26th International Conference on Data Engineering Workshops (ICDEW 2010)*. IEEE, 221–226.

[24] Andreas Thalhammer, Nelia Lasierra, and Achim Rettinger. 2016. Linksum: using link analysis to summarize entity data. In *International Conference on Web Engineering*. Springer, 244–261.

[25] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*. PMLR, 2071–2080.

[26] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *International Conference on Learning Representations* (2018). https://openreview.net/forum?id=rJXMpikCZ accepted as poster.

[27] Ziming Wang, Jun Chen, and Haopeng Chen. 2021. EGAT: Edge-Featured Graph Attention Network. In *Artificial Neural Networks and Machine Learning – ICANN 2021*. Springer International Publishing, Cham, 253–264.

[28] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 28.

[29] Dongjun Wei, Shiyuan Gao, Yaxin Liu, Zhibing Liu, Longtao Huang, and Songlin Hu. 2018. MPSUM: Predicate-Based Matching for RDF Triples with Application to LDA. In *EYRE@CIKM*.

[30] Dongjun Wei, Yaxin Liu, Fuqing Zhu, Liangjun Zang, Wei Zhou, Jizhong Han, and Songlin Hu. 2019. ESA: Entity Summarization with Attention. In *EYRE@CIKM*.

[31] Dongjun Wei, Yaxin Liu, Fuqing Zhu, Liangjun Zang, Wei Zhou, Yijun Lu, and Songlin Hu. 2020. AutoSUM: Automating Feature Extraction and Multi-user Preference Simulation for Entity Summarization. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 580–592.

[32] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. http://arxiv.org/abs/1412.6575

[33] Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. 2017. Graph-based Neural Multi-Document Summarization. In *Proceedings of the 21st Conference on Computational Natural Language*

[5] http://www.bioasq.org/

Learning (CoNLL 2017). Association for Computational Linguistics, Vancouver, Canada, 452–462. https://www.aclweb.org/anthology/K17-1045

[34] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. 2006. Learning with Hypergraphs: Clustering, Classification, and Embedding. In *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*. MIT Press, 1601–1608.