Semantic Novelty Detection in Natural Language Descriptions

Nianzu Ma[†], **Alexander Politowicz**[†], **Sahisnu Mazumder**[†],
Jiahua Chen[†], **Bing Liu**[†], **Eric Robertson**[♮], **Scott Grigsby**[♮]
[†]Department of Computer Science, University of Illinois at Chicago, USA
[♮]PAR Government Systems Corporation, USA
jingyima005@gmail.com, politow2@uic.edu
{sahisnumazumder,jiahuaqy}@gmail.com
liub@uic.edu, {eric_robertson,Scott_Grigsby}@partech.com

Abstract

This paper proposes to study a fine-grained semantic novelty detection task, which can be illustrated with the following example. It is normal that a person walks a dog in the park, but if someone says "*A man is walking a chicken in the park*," it is novel. Given a set of natural language descriptions of normal scenes, we want to identify descriptions of novel scenes. We are not aware of any existing work that solves the problem. Although existing novelty or anomaly detection algorithms are applicable, since they are usually topic-based, they perform poorly on our fine-grained semantic novelty detection task. This paper proposes an effective model (called GAT-MA) to solve the problem and also contributes a new dataset. Experimental evaluation shows that GAT-MA outperforms 11 baselines by large margins.

1 Introduction

Novelty or anomaly detection has been an important research topic since 1970s (Barnett and Lewis, 1994) due to numerous applications (Chalapathy et al., 2018; Pang et al., 2021). Recently, it has also become important for natural language processing (NLP). Many researchers have studied the problem in the text classification setting (Fei and Liu, 2016; Shu et al., 2017; Xu et al., 2019; Lin and Xu, 2019; Zheng et al., 2020). However, these text novelty classifiers are mainly *coarse-grained*, working at the document or topic level. Given a text document, their goal is to detect whether the text belongs to a known class or unknown class.

This paper introduces a new text novelty detection problem - *fine-grained semantic novelty detection*. Specifically, given a text description d, we detect whether d represents a semantically novel fact or not. This work considers text data that describe scenes of real-world phenomena in natural language (NL). In our daily lives, we observe different real-world phenomena (events, activities, situations, etc.) and often describe these observations (referred as "scenes" onwards) in NL to others or write about them. It is quite natural to observe scenes that we have not seen before (i.e., novel scenes). For example, it is a common scene that "*A person **walks a dog** in the park*", but if someone says "*A man is **walking a chicken** in the park*", it is quite unexpected and novel. Detecting such semantic novelty requires complex conceptual and semantic reasoning over text and thus, is a challenging NLP problem. Note that conceptually, the judgement of the novelty of a scene is subjective and might differ from person to person. However, there are some scenes for which a majority of people have agreement about their novelty. A good example of such majority-view of novelty is the widely-spread meme pictures on social media, which contain novel interactions between objects. In this work, we restrict our research to this majority-based view of novelty and leave the personalized novelty view angle for the future work.

In this work, we leverage the captions of images from popular datasets like COCO, Flickr, etc., to build a semantic novelty detection dataset (Sec. 3),[1] where we consider an image as a scene and the corresponding image captions as different NL descriptions of the scene. Detecting text describing semantically novel observations have many applications, e.g., recommending novel news, novel images & videos (based on their text descriptions), social media posts and conversations. The problem of semantic novelty detection is defined as follows.

Problem Definition: Given a set of natural language descriptions $\mathcal{D} = \{d_1, d_2, ...d_n\}$ of common scenes, build a model $\mathcal{M}$ using $\mathcal{D}$ to score the semantic novelty of a test NL description d' with respect to $\mathcal{D}$, i.e., classifying d' into one of the two classes {*NORMAL*, *NOVEL*}. "*NORMAL*" means that d' is a description of a common scene and "NOVEL" means d' is a description of a se-

[1]No image is used in this work.

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 866–882
November 7–11, 2021.

mantically novel scene. As the detection model $\mathcal{M}$ is built only with "*NORMAL*" class data, the task is an *one-class text classification problem*.

We are unaware of any existing work that can effectively solve this problem. Although existing novelty/anomaly detection and one-class classification algorithms are applicable, since they are coarse-grained or topic-based, they perform poorly on our task (see Sec. 5). Note that although we focus on solving the problem of semantic novelty detection of NL descriptions of scenes, the proposed task and solution framework are generally applicable to other applications.

This paper proposes a new technique, called GAT-MA (***G**raph **At**tention network with Max-**M**argin loss and knowledge-based contrastive data **A**ugmentation*) to identify NL description sentences of novel scenes. Since our task is at the *sentence level* and *fine-grained*, we exploit Graph Attention Network (GAT) on the parsed dependency graph of each sentence, which fuses both semantic and syntactic information in the sentence for reasoning with the internal interactions of entities and actions. To enable the model to capture long-range interactions, we stack multiple layers of GATs to build a deep GAT model with multi-hop graph attention. We also create the pseudo novel training data based on the given normal training data through contrastive data augmentation. Thus, GAT-MA is trained with the given original normal scene descriptions and the augmented pseudo novel scene descriptions (Sec. 4).

GAT-MA is evaluated using our newly created **N**ovel **S**cene **D**escription **D**etection (**NSD2**) Dataset. The results show that GAT-MA outperforms a wide range of latest novelty or anomaly detection baselines by very large margins. Our main contributions are as follows:

1. We propose a new task of *semantic novelty detection in text*. Whereas the existing work focuses on coarse-grained document- or topic-level novelty, our task requires fine-grained sentence-level semantic & syntactic analysis.
2. We propose a highly effective technique called GAT-MA to solve the proposed semantic novelty detection problem, which is based on GAT with dependency parsing and knowledge-based contrastive data augmentation.
3. We create a new dataset called NSD2 for the proposed task. The dataset can be used as a benchmark dataset by the NLP community.

2 Related Work

Our work is closely related to anomaly, outlier or novelty detection. Earlier approaches include *one-class SVM* (OCSVM) (Schölkopf et al., 2001; Manevitz and Yousef, 2001) or *Support Vector Data Description* (SVDD) (Tax and Duin, 2004). In recent years, deep learning approaches dominated. Erfani et al. (2016) and Ruff et al. (2018) learned features using deep learning and then applied OCSVM or SVDD to build one-class classifiers. Many recent approaches are based on auto-encoders (You et al., 2017; Abati et al., 2019; Chalapathy and Chawla, 2019), GAN (Perera et al., 2019; Zheng et al., 2019), neural density estimation (Wang et al., 2019), multiple hypothesis prediction (Nguyen et al., 2019), robust mean estimation (Dong et al., 2019) and regularization (Hu et al., 2020). See the surveys (Chalapathy and Chawla, 2019; Pang et al., 2021) for more details. Our GAT-MA is based on stacked graph attention neural networks, parsing and data augmentation.

Novelty detection has also been studied in out-of-distribution (OOD) detection (Fei and Liu, 2016; Fei et al., 2016; Liang et al., 2018; Shu et al., 2018; Erfani et al., 2017; Xu et al., 2019). However, these methods work in the multi-class classification setting. Our work focuses on one-class classification.

Our work is also related to document or sentence topical novelty detection (Dasgupta and Dey, 2016; Ghosal et al., 2018; Nandi and Basak, 2020; Jo et al., 2020; Zhang et al., 2003; Ru et al., 2004; Li and Croft, 2005; Zhang and Tsai, 2009). These tasks differ from our problem setting as we focus on fine-grained semantic novelty detection.

Our work is also related to *semantic plausibility* (SPLA) and *selectional preference* (SPRE). SPLA is concerned with whether an event is plausible, and SPRE is about the "typicality" of an event. For SPLA, existing models employ pretrained language models (Porada et al., 2019) and manually elicited entity property knowledge (Wang et al., 2018) to model physical plausibility in the supervised setting. Other related work includes creating datasets with plausibility ratings (Keller and Lapata, 2003) and dealing with multi-event inference (Zhang et al., 2017; Sap et al., 2019). For SPRE, the early works include (Resnik, 1996; Clark and Weir, 2001; Erk and Padó, 2010; Bergsma et al., 2008; Ritter et al., 2010; Ó Séaghdha, 2010; Van de Cruys, 2009). The performance is improved by neural networks (Van de Cruys, 2014; Dasigi and Hovy,

2014; Tilk et al., 2016). Our work is different: (1) Conceptually, SPLA and SPRE are related but different from novelty, (2) they are mostly based on structured Subject-Verb-Object triples, rather than natural language sentences, and (3) they use fully labeled data (Dasigi and Hovy, 2014) while we do novelty detection with only normal data in training.

The work of commonsense reasoning is remotely related to our work. Existing works build multi-choice commonsense reasoners (Zellers et al., 2018, 2019), study the commonsense knowledge contained in language models (Davison et al., 2019; Trinh and Le, 2019, 2018) and knowledge graph (Bosselut et al., 2019), and build new datasets for better evaluation (Wang et al., 2020a). Several researchers also investigated physical commonsense reasoning (Bagherinezhad et al., 2016; Forbes and Choi, 2017; Wang et al., 2017; Bisk et al., 2020) and affordance of entities (Forbes et al., 2019). They do not perform novelty detection.

Our work is also related to trivia fact mining (Merzbacher, 2002; Ganguly et al., 2014; Gamon et al., 2014; Prakash et al., 2015; Fatma et al., 2017; Mahesh and Karanth; Tsurel et al., 2017; Niina and Shimada, 2018; Korn et al., 2019; Kwon et al., 2020). However, trivia is more related to interestingness. Some trivia facts are interesting because they are rare, but not necessarily novel. Existing works use labeled training data for learning, or rely on Wikipedia structure to retrieve interesting facts using information retrieval methods (Tsurel et al., 2017; Kwon et al., 2020). We have only normal data but not novel data.

Our proposed model learns text representation using a Graph Neural Network and leveraging dependency parsing. Other works in NLP that use Graph Neural Networks and dependency structures include (Huang and Carley, 2019; Ma et al., 2020; Guo et al., 2019; Wang et al., 2020b; Pouran Ben Veyseh et al., 2020; Xiao and Zhou, 2020), etc. But they solve different problems, such as sentiment analysis and argument mining. Their approaches are also different from ours and do not do novelty detection.

3 Dataset Collection and Annotation

As there is no semantic novelty detection dataset available for text, we build a new dataset. As our proposed task requires learning of *latent* semantic knowledge in text, such as capturing the interaction among entities and verbs (e.g. “person" and “food" are related to each other by verb “cook"); the actions (verbs) that an entity can support (e.g., only a person can perform action “cook"); actions an entity can be applied on (e.g. “cook" can be applied on entity “vegetables"), etc., we aim to build a corpus rich in such knowledge. Text data like news articles, social media posts, reviews, etc., generally contain such knowledge in low density and thus, are not very suitable. Instead, we leverage image captions to built our dataset, which we found to be suitable for our task.

Image caption data collection. We found that the captions of non-iconic images (depicting multiple objects and their interactions) meet the aforementioned dataset requirements. We chose three popular benchmark image caption datasets: COCO (Chen et al., 2015; Lin et al., 2014), Flickr30k (Plummer et al., 2015) and Visual Genome (Krishna et al., 2017) to build our dataset. COCO consists of 616,435 captions of Flickr images. Flickr30k contains 158,915 captions about people and animals, and Visual Genome contains 5.4 million captions describing interactions among various objects. To ensure we have a diverse dataset to learn interactions among entities and verbs, we merge the 3 datasets into one large dataset.

NSD2 dataset preparation. Given the merged NL caption dataset, we proceed to build our proposed NSD2 dataset as follows. We consider the captions from the NL caption dataset as normal or common scene descriptions. As our proposed GAT-MA model uses only “NORMAL" class data, we build our **training dataset** involving **only normal scene descriptions** and compile a **test dataset** having scene descriptions involving **both “NORMAL" and “NOVEL"** classes.

Due to budgetary constraints, we cannot evaluate on all verbs. We selected 20 verbs (see Appendix Sec. A) frequently used in the NL caption dataset and built our training and test dataset with scene descriptions involving these 20 verbs. For training, we extract the captions from the merged set that contain any of the 20 verbs as the “NORMAL" class text examples. For test dataset preparation, we employ human annotators to write NL scene descriptions involving both “NORMAL" and “NOVEL" classes (discussed below).

Test dataset preparation. The test dataset is prepared by 5 volunteer graduate students with advanced level of English as crowd workers. We divide the task into 20 small subtasks, one verb

Table 1: NSD2 dataset statistics. NR (NV) denotes NORMAL (NOVEL) class. "description length" denotes # words.

	Training	Test
# instances (descriptions)	202,681 (NR)	2000 (NR), 2000 (NV)
Avg. description length	11.25	11.10

for each subtask. For each substask, the designated worker is asked to write at least 100 normal and 100 novel scene descriptions from scratch for this verb. For training the workers, each of them is asked to write 25 normal and 25 novel sentences for a verb and then we check these sentences and give them feedback. Any disagreements are discussed. After the training session, each subtask is carried out by each worker independently. The workers are unaware of the proposed model. After initial writing of each subtask is done, the scene descriptions are assigned to other four workers (who are not the writer) to label them as normal or novel. If the consensus (majority judgment) is the same as the original writer's label of the scene description, it means this scene description's label aligns with the majority view of novelty. If the majority judgement is not the same as the original writer's label, this scene description is discarded. Then the worker is asked to write more and iterate the above voting process until 100 normal and 100 novel scene descriptions are collected for this verb.

Table 1 shows the summary of our NSD2 dataset statistics. More detailed statistics regarding training data statistics for each verb and description token number are provided in Appendix Sec.A.

4 The Proposed GAT-MA Model

The proposed GAT-MA model consists of two main components: **(i) Knowledge-based Contrastive Data Generator (CDG)**, and **(ii) Text Semantic Novelty Scorer (SNS)**. Given a set of NL descriptions $\mathcal{D}^{tr} = \{d_1, d_2, ..., d_n\}$ of normal scenes in the training data, CDG dynamically generates *pseudo-novel descriptions* by perturbing the normal scene descriptions in $\mathcal{D}^{tr}$ utilizing the lexical knowledge base WordNet[2] (Fellbaum, 2010). The normal descriptions in $\mathcal{D}^{tr}$ are augmented with these pseudo-novel descriptions (used as NOVEL class examples in training) to learn a SNS.

The SNS is a deep GAT model that learns to score an input text to measure its semantic novelty with respect to $\mathcal{D}^{tr}$. To capture the semantic and syntactic information in an input text d, GAT-MA parses d into a dependency graph and feeds the graph enriched with additional word-level features to the SNS, which is then trained to assign higher score to a normal scene description compared to that of a novel one.

4.1 Knowledge-based Contrastive Data Generator (CDG)

We propose to use the lexical knowledge base WordNet to help generate contrastive instances to the normal scene descriptions in $\mathcal{D}^{tr}$. These contrastive instances serve as pseudo-novel data and enable supervised learning of the Text Semantic Novelty Scorer (SNS). WordNet contains rich taxonomy of words and thus, is beneficial to our semantic novelty detection task.

In our generator, a **knowledge-based misfit sampler** $S_{misfit}(.)$ is the key component. Given a normal scene description $d \in \mathcal{D}^{tr}$, $S_{misfit}(e)$ [here, e is an entity, either a noun or a noun phrase] samples an entity e' that is semantically distant from e in the WordNet. We use Wu-Palmer Similarity (Wu and Palmer, 1994) to measure the semantic distance between e and e'. We randomly sample e' from WordNet such that the similarity score between e and e' is less than 0.9 (an empirically set threshold). Next, since e' is semantically distant from e, e' is a misfit in original description d. e is replaced with e' in description d to generate a pseudo-novel description. For example, "*a **man** is driving a car*" describes a normal scene. It is commonsense that the subject for verb "drive" should be a person. Any thing outside of the category introduces novelty, e.g., "*a **dog** is driving a car*".

When replacing the entity e, the choice of e is also critical. For our task, we focus on three novelty aspects in a given description: (1) what actions an entity can perform, (2) what actions an entity can be applied to, and (3) how several entities interact with each other. In the interactions between entities and verbs, verbs are the core of these interactions. Thus, we only replace entities that are syntactically related to a verb to create pseudo-novel descriptions. We refer to the verb of interest in d as the *target verb*, which is used later in Sec. 4.2. We include the details for finding and extracting entities syntactically related to the target verb in the Appendix Sec.B.

Note, the novel scene description d' generated by the perturbation is contrastive to the original description d. We dynamically generate one (empiri-

[2]https://wordnet.princeton.edu/

cally set) pseudo-novel description for each normal description in $\mathcal{D}^{tr}$ in every training epoch.

4.2 Text Semantic Novelty Scorer (SNS)

The recent progress of employing GAT (Velickovic et al., 2018) on text data (Huang and Carley, 2019; Ma et al., 2020; Guo et al., 2019) has shown the advantage of explicitly combining syntactic structure (dependency parse graph) and word-level semantics for fine-grained text analysis, such as aspect-level sentiment analysis and argument mining. Because our task is inherently a fine-grained semantic reasoning task, we build SNS based on GAT. GAT fuses the graph-structured information and node features and employs masked self-attention layers to allow a node to attend to its neighborhood features and learn different attention weights for different neighboring nodes for graph representation learning. More details can be found in Appendix Sec. D.

4.2.1 Input Representation

We use a dependency parser (Chen and Manning, 2014) to convert an input scene description d into a dependency parse graph. For a description $d = \{w_1, w_2, ...w_n\}$, a word w_i corresponds to a node n_i in the graph. The node feature of n_i is a word embedding vector: $X_i \in \mathbb{R}^F$. F is the word embedding size. Since a description contains n words, the input node feature matrix is $\boldsymbol{X} \in \mathbb{R}^{n \times F}$.

4.2.2 Enriching Entity Word Embeddings with Hypernym Information

We consider a noun or a noun phrase in d as an **entity** if it exists in the WordNet. And we refer the word(s) comprising the entity as **entity word**(s) and the corresponding word embedding(s) as **entity word embedding**(s) onwards. Intuitively, the hypernym information of entities is beneficial to our task. Consider a normal description, "*a golden retriever is chasing a flying frisbee*". One of the hypernym chains of the entity "*golden retriever*" in WordNet is: {golden retriever}[3] $\Rightarrow \ldots \Rightarrow$ {dog, domestic dog, Canis familiaris} $\Rightarrow \ldots \Rightarrow$ {carnivore} $\Rightarrow \ldots \Rightarrow$ {mammal, mammalian} $\Rightarrow \ldots$ {entity}. This hypernym chain tells that *golden retriever* is a breed of dog. If we leverage the hypernym information, the model can not only learn that one specific breed of dog like "*golden retriever*" can chase a frisbee, but also generalize to other breeds of dogs as well. Additionally, this hypernym chain also contains other commonsense knowledge such as "*dogs eat meat*", since dogs belong to the category "*carnivore*".

We perform the following three steps to incorporate hypernym features into GAT-MA:

Step-1. Candidate Entity Set Extraction. We incorporate hypernym features to entities that are syntactically related to the target verb in a description. We call these entities the candidate entities onwards. Given an input description d, this step extracts the candidate entities from d using a rule-based extractor that leverages dependency parsing and POS tagging information. Details of the method can be found in Appendix Sec. B. Considering the aforementioned example, the candidate entities are "*golden retriever*" and "*frisbee*" and the target verb is "*chase*".

Step-2. Obtaining Hypernym Name Set from WordNet. Given a entity e, the Hypernym Name Set of e is the set of synset names of hypernyms of e in the WordNet. Considering the entity "*golden retriever*", we obtain its **Hypernym Name Set** from WordNet as follows:

1. **Obtain the synet of the entity**. The concept of *hypernym* is defined between synsets in the WordNet. The word sense of an entity e defined in the description context corresponds to a synset in the WordNet. Ideally, a Word Sense Disambiguation (WSD) model should be employed to tag this entity with an appropriate synset. We have tried state-of-the-art WSD models, and found them not working well for our dataset. On analysis, we found that choosing the first sense of the entity works better. Note that, according to the WordNet documentation[4], "*Senses in WordNet are generally ordered from most to least frequently used, with the most common sense numbered 1.*" which conforms to our findings.

2. **Find a complete Hypernym Synset Set.** With the chosen synset of the entity, we recursively collect the set of all hypernym synsets from the WordNet. For instance, given the entity "*golden retriever*", the set of synsets in all hypernym chains originating from {golden retriever} synset to {entity} synset in the WordNet hypernym hierarchy forms the Hypernym Synset Set of "*golden retriever*".

[3] We show a synset in the format of a list of lemma names to make a synset more informative to demonstrate.

[4] https://wordnet.princeton.edu/documentation/wndb5wn

3. **Filter General Hypernym Synsets.** In practice, when compiling the entity hypernym information, we do not consider the whole Hypernym Synset Set for that entity because some hypernyms are too general to contribute useful knowledge for our task. Thus, we manually collect a set of synsets that are too general and remove them from the complete Hypernym Synset Set of the entity. The 24 general sysnets are given in Appendix Sec. C.

4. **Get Hypernym Name Set.** An hypernym synset contains a set of lemma names. E.g. given a hypernym synset - Synset('dog.n.01') of entity "*golden retriver*", "*dog*", "*domestic dog*", "*Canis familiaris*" are the lemma names. We obtain the Hypernym Name Set of an entity by collecting all lemma names from all synsets in the Hypernym Synset Set of the entity.

Step-3. Construction of Hypernym Feature Vector. A Hypernym Feature Vector is created for each entity based on its Hypernym Name Set and is computed as the pointwise addition of all Hyernym Name Embeddings, one for each Hyernym Name in the Hypernym Name Set of the entity. We use two types of Hypernym Name Embeddings as follows:

- **GloVe-based Hyernym Name Embedding.** For a single-word hypernym name, the Hyernym Name Embedding is the corresponding GloVe word embedding. For a multi-word Hypernym Name, it is computed as the average of GloVe embeddings of the words in the Hypernym Name.
- **BERT based Hypernym Name Embedding.** Since BERT produces contextual embedding for each word, the input of BERT should contain the context information. Given an input description, we replace the entity in the description with the Hypernym Name and feed this description into BERT. Because BERT tokenizer segments word into word pieces (subword tokens), we average the embeddings of all word pieces corresponding to this Hypernym Name to obtain the final Hypernym Name Embedding.

The Hypernym Feature Vector F^{hyper} is calculated as: $F^{hyper} = \sum_{k=1}^{M} X_k^{hyper}$, where X_k^{hyper} is the embedding of the k^{th} Hypernym Name in Hypernym Name Set of an entity, and M is the size of the Hypernym Name Set.

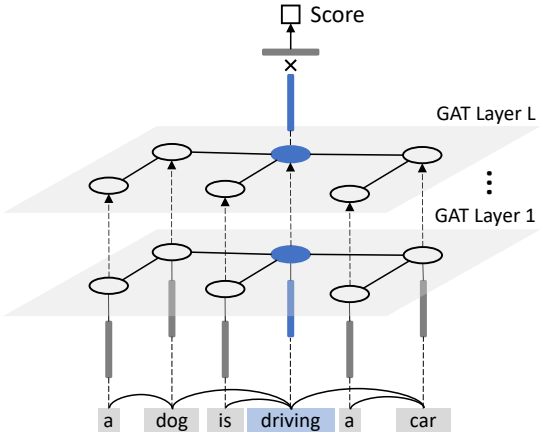

Figure 1: Working of GAT-MA on an input text.

4.2.3 Modeling Dependency using Deep GAT

We observe that the dependency parse graph of description d contains rich syntactic information that is beneficial to explicitly learn the interactions between entities and actions in a scene description, especially long range interactions. For a novel description like "*a monkey with a white beard and brown hair is driving a car down the street*", the interaction among *monkey*, *drive* and *car* makes it semantically novel. Note that, entity "*monkey*" and verb "*drive*" have a sequential word distance of 9 making it difficult for a sequential representation learning method to model the interaction. In contrast, "*monkey*" and "*drive*" are only one hop away in the dependency parse tree.

In addition, we find that for these three key words, "*drive*" is the parent of both "*monkey*" and "*car*" in the original directed dependency graph. To encourage interactions between them and allow the semantic information to flow freely in the dependency graph structure during training, we simplify the original directed dependency graph into an undirected graph. Importantly, the GAT model is trained not to attend to all neighbors of a given node equally. The attention weights to neighbors are trained to give higher weights to those nodes more useful for the task.

The input-output for a single GAT layer is summarized as $\boldsymbol{H}^{out} = GAT(\boldsymbol{X}, \boldsymbol{A}; \Theta)$. The input is $\boldsymbol{X} \in \mathbb{R}^{n \times F}$ and the output is $\boldsymbol{H}^{out} \in \mathbb{R}^{n \times F'}$, where n is the number of nodes, F is the node feature size, F' is GAT hidden size, and the dependency graph structure is encoded into $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ which is the adjacency matrix of the graph.

In a single GAT layer, a word or an entity in a graph only attends over the local information from 1-hop neighbors. To enable the model to cap-

ture long-range interactions between entities and actions, we stack L layers to make a *deep* model, which allows information from L-hops away to propagate into this word.

As illustrated in Figure 1, the stacking architecture is represented as $\boldsymbol{H}^{l+1} = GAT(\boldsymbol{H}^l, \boldsymbol{A}; \Theta^l)$, $l \geq 0$, $\boldsymbol{H}^0 = \boldsymbol{X}\boldsymbol{W}_0 + \boldsymbol{b}_0$. The output of the GAT layer l, $\boldsymbol{H}^l_{out} = GAT(\boldsymbol{H}^l, \boldsymbol{A}; \Theta^l)$, is the input for layer $(l+1)$, denoted by $\boldsymbol{H}^{l+1}$. $\boldsymbol{H}^0$ is the initial input. $\boldsymbol{W}_0 \in \mathbb{R}^{F \times F'}$ and $\boldsymbol{b}_0$ are the projection matrix and bias vector. For a L layer GAT-MA model, the output of the final layer is $\boldsymbol{H}^L_{out} \in \mathbb{R}^{n \times F'}$.

For our task, we are concerned with interactions of verbs and entities. As mentioned in Sec. 4.1, when perturbing the normal descriptions, we only replace the entities that are syntactically related to a verb in the dependency graph. This verb is our target verb. Any novelty introduced in the description due to the replacement is related to this verb. If a description contains multiple verbs, the target verb of an entity is the one which is close to it along the dependency parse graph.

We use a mask layer $\boldsymbol{m}$ to fetch the output embedding for this target verb v_i from GAT: $\boldsymbol{h}_{v_i} = \boldsymbol{m}\boldsymbol{H}^L_{out}$, where $\boldsymbol{m} \in \mathbb{R}^{1 \times n}$ is a one-hot vector indicating the position of the target verb. Next, we use a feed-forward layer to project $\boldsymbol{h}_{v_i}$ into a semantic novelty score. We denote the score function of SNS by $S(d)$ for the input description d.

Training. GAT-MA is trained end-to-end by minimizing a max-margin ranking objective, as given below -

$$\mathcal{L} = \sum_{d \in \mathcal{D}^{tr}} \sum_{d' \in D'} max\{S(d') - S(d) + 1, 0\} \quad (1)$$

where, $\mathcal{D}^{tr}$ is the set of the normal descriptions, $d' \in D'$ is the pseudo-novel description corresponding to $d \in \mathcal{D}^{tr}$. $\mathcal{L}$ encourages the score $S(d)$ of normal description d to be higher than $S(d')$ for a pseudo-novel description d'.

5 Experiments

5.1 Experiment Setup

For dataset details, please refer to Sec. 3. Appendix has additional information about the data and model implementation details.[5]

[5]The code and the annotated dataset are released at: https://github.com/NianzuMa/semantic-novelty-detection-in-natural-language-descriptions

Baselines. We compare GAT-MA with three categories of baselines: (1) four language model based novelty detection models, (2) seven one-class classification models, (3) other models based on different text encoders and loss functions (see Sec. 5.2). All the results in this section are the average of five runs with different seeds. The results are statistically significant with $p < 0.001$.

A trained language model (LM) can be intuitively used as a novelty detection model due to the following reasons: (1) When training a LM on normal scene descriptions, the model minimizes the perplexity of the training data by maximizing the likelihood of each word appearing in its context. In this way, it indirectly learns the semantic meaning of words and sentences. (2) Each LM trained on normal descriptions can output the probability of each word in a description appearing in its context. Thus a sentence probability can be calculated from the list of word probabilities. We have tried various ways of calculating the sentence score from the word probability list, such as arithmetic mean, geometric mean, harmonic mean, and multiplication of all word probabilities and found harmonic mean to be the best choice. We use **N-gram**, the bag of words LM, $N \in \{1, 2, 3, 4, 5\}$ ($N = 1$ gives the best result), **LSTM** (Hochreiter and Schmidhuber, 1997), **BERT** (Devlin et al., 2019), **GPT-2** (Radford et al., 2019) as our LM baselines. The results are listed in Table 2.

For general one-class classification models, most of them only work on images. We modified the related components of the models to make them suitable for text data. More details regarding model modification and parameter setting are provided in Appendix Sec. F. The following 7 baselines are compared: (1) **DSVDD** (Deep SVDD) (Ruff et al., 2018): a recent one-class classifier, which is the deep learning version of SVDD (see Sec. 2). (2) **ICS** (Schlachter et al., 2019): a recent one-class classification method trained on one class of training data that is split into two subsets: typical and atypical. (3) **OCGAN**(Perera et al., 2019): a latest one-class anomaly detection method based on GAN. (4) **VAE** (Kingma and Welling, 2014): the variational auto-encoder. (5) **OCSVM** (Schölkopf et al., 2001): the classic SVM method for one-class classification (see Sec. 2). (6) **iForest** (Liu et al., 2008): a classic ensemble method based on random unsupervised trees. (7) **HRN** (Hu et al., 2020): the latest model based on a holistic regular-

Table 2: Comparison of baselines and our proposed model (based on AUC score)

Language model based model				General One-class classifier							Proposed
Ngram	LSTM	BERT	GPT-2	OCSVM	iForest	VAE	DSVDD	ICS	OCGAN	HRN	**GAT-MA**
76.76	77.95	82.13	77.87	68.07	50.55	51.43	54.89	56.15	50.80	56.83	**89.22**

ization. We could not compare with another latest baseline CSI (Tack et al., 2020) as it is based on various image transformations. We do not compare with out-of-distribution (OOD) detection methods as they require multiple classes to learn.

Experiments settings. In general, we conduct experiments using various word and sentence embeddings, such as GloVe[6] (Pennington et al., 2014), BERT[7] (Devlin et al., 2019) and InferSent (Conneau et al., 2017; Bowman et al., 2015). We only show the best results in Table 2. The detailed hyperparameter settings for GAT-MA and baseline models are included in the Appendix Sec. E and F.

Evaluation Metrics. Following the existing novelty/anomaly detection literature (Chalapathy and Chawla, 2019; Pang et al., 2021), we only produce a score function and ignore the binary decision problem and use AUC (Area Under the ROC curve) as the evaluation metric. All compared models are trained with only normal scene descriptions.

5.2 Results and Analysis

Baseline Comparison. Table 2 shows the predictive performance comparison of the baselines and our proposed model GAT-MA. Note, GAT-MA is our proposed model using BERT embedding and enhanced with hypernym embedding features. From Table 2, we conclude the following:

(1) All general one-class classifiers perform poorly on our task. Even the reported state-of-the-art model HRN gives AUC score of only 56.89. We have tried various ways to produce the description embedding as the input feature for these models, such as (a) averaging all words' GloVe embeddings, (b) feeding the description into BERT and using the first token [CLS]'s embedding as the sentence embedding, (c) feeding the description into BERT and averaging all output tokens' embeddings as the sentence embedding, and (d) feeding the description into the pre-trained sentence embedding extractor InferSent to produce the sentence embedding. However, none of these options give good performances. These one-class classifiers perform well on image data because images of a given class (e.g., in the MNIST dataset) contains images with very similar latent representations. Thus, auto-encoder and GAN-based models can learn latent representations for all instances in an image class very close to each other in the latent space. In contrast, our normal scene descriptions have many topics and it's hard for them to learn latent representations that are close to each other in the latent space.

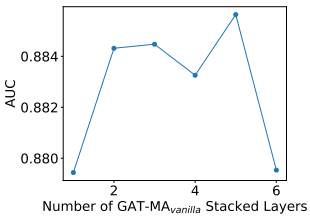

Figure 2: Effects of the number of layers in GAT-MA$_{\text{vanilla}}$

(2) Language model-based methods are in general better than one-class classifiers because they, in some sense, do not try to learn a latent representation, but exploit the sequential and semantic information of the input text to produce word probabilities. Thus, they are comparatively more effective in fine-grained semantic novelty detection. However, they still perform much worse than GAT-MA as they mainly learn the word distribution in the normal description data but do not explicitly capture the interaction of entities and verbs.

In summary, GAT-MA outperforms all baselines by large margins and is more effective for our proposed task. Below, we discuss ablation and additional experiments.

Effects of word embedding and hypernyms. In Table 4, GAT-MA$_{\text{vanilla}}$ is our proposed model using BERT embedding without being enhanced with the hypernum features. GAT-MA$_{\text{GloVe}}$ is our proposed model using GloVe embedding without being enhanced with hypernym features. Comparing the results of GAT-MA$_{\text{GloVe}}$ and GAT-MA$_{\text{vanilla}}$, we can see that BERT embedding contains richer semantic knowledge which is more beneficial to our task compared to using GloVe embedding. It is also interesting to see that when GAT-MA$_{\text{vanilla}}$ is enhanced with hypernym embedding feature (noted as GAT-MA), it improves the AUC score from 88.12 to 89.22. It means that hypernym features can help our model generalize better.

[6]We use glove.840B.300d in our experiments

[7]We use the BERT model "bert-base-uncased" as text encoder. We expect that using larger transformer embeddings leads to better results. But due to the limitation of our computing resources, we have to use this base BERT model.

Table 3: Some descriptions predicted wrongly by BERT_{MM} but correctly by $\text{GAT-MA}_{\text{MM}}$

	Text	Label
1	a monkey with glasses is cooking food on a stovetop in a kitchen.	Novel
2	a couple of seal dogs carry their surfboard across the beach.	Novel
3	a giant panda in a white smock prepares to cut the hair of an older balding gentleman in front of a case holding several hair supplies.	Novel
4	an adult is walking on the sidewalk in St. Luis.	Normal
5	a guy eats food on a table in front of a food shop on the street while a passerby walks by.	Normal
6	a group of people stands around are drinking some vermouth.	Normal

Table 4: Effect of using embedding type and hypernym feature based on AUC score

$\text{GAT-MA}_{\text{GloVe}}$	$\text{GAT-MA}_{\text{vanilla}}$	GAT-MA
84.42	88.12	89.22

Table 5: Comparison of BERT and GAT-MA variants based on cross-entropy (CE) and max-margin (MM) loss function based on AUC scores

BERT_{CE}	BERT_{MM}	$\text{GAT-MA}_{\text{CE}}$	$\text{GAT-MA}_{\text{MM}}$
82.09	87.41	83.80	88.12

Effects of model depth. From Figure 2, we see that increasing the number of stacked layers from 1 to 5 improves the performance of $\text{GAT-MA}_{\text{vanilla}}$. When the number of stacked layers higher than 5, the performance drops. This is because most of the interaction between entities and actions near to each other in the dependency parse graph. Stacking 5 layers is enough and more stacked layers will not help but hurt the performance.

Effects of using max-margin ranking loss. Table 5 compares fine-tuned BERT and GAT-MA variants in terms of the use of loss functions in model training. Here, $[\cdot]_{\text{CE}}$ denotes the model using the cross entropy loss for training and $[\cdot]_{\text{MM}}$ denotes the model using the max-margin loss as proposed in Sec. 4.2.3. From Table 5, we see that CE variants are weaker than MM variants for both BERT and GAT-MA. Both $\text{GAT-MA}_{\text{CE}}$ and $\text{GAT-MA}_{\text{MM}}$ use BERT embeddings without the hypernym feature.

Effects of using dependency parse structure. Table 5 shows that BERT_{MM} not directly using any syntactic features easily fail on examples dissimilar to training data in terms of word distribution. However, $\text{GAT-MA}_{\text{MM}}$ performs better by explicitly modeling the dependency parse structure. This means that modeling dependency parse structure is beneficial to capturing the interactions between entities and actions in our task. Some descriptions predicted wrongly by BERT_{MM} but correctly by $\text{GAT-MA}_{\text{MM}}$ are shown in Table 3.

6 Error Analysis

The AUC score in (.) for each verb is as follows: pull (0.99), carry (0.99), push (0.99), drive (0.97), travel (0.97), hit (0.95), throw (0.95), kick (0.94), climb (0.94), look (0.93), build (0.93), cook (0.92), walk (0.92), ride (0.87), fly (0.84), cut (0.82), swim (0.81), jump (0.73), drink (0.73), and eat (0.69).

We carried out error analysis on our test data and found that the errors are mainly due to the following factors. The first factor is the pretrained word embeddings' quality. The quality of the word embedding is critical for GAT-MA to effectively do reasoning. GAT-MA makes mistakes when the pretrained word embedding is not of good quality. For example, the "*talapoin*" in "*the talapoin at the zoo is leaning down to drink some water*". The second factor is the limitation of knowledge acquired by GAT-MA during training. GAT-MA relies on the taxology information in WordNet to generate contrastive novel descriptions during training. However, sometimes the reasoning of novel description requires more complex world knowledge. For examples, "two kids are sitting in the bar drinking spirit" is novel and requires knowledge that kids is not old enough to drink any alcohol. Another example "A dog is eating onions on the ground" is novel and requires the world knowledge that onions is poisonous to dogs[8].

7 Conclusion

Novelty detection is an important problem because anything novel is of interest. This paper proposed a semantic novelty detection problem and designed a graph attention network based approach (called GAT-MA) exploiting parsing and data augmentation to solve the problem. As there is no existing evaluation dataset for the proposed task, an evaluation dataset has been created. Experimental comparisons with a wide range of baselines showed that GAT-MA outperforms them by very large margins.

[8] https://en.wikipedia.org/wiki/Dog_health

Acknowledgments

This work was supported in part by a DARPA Contract HR001120C0023, two National Science Foundation grants IIS-1910424 and IIS-1838770, and a research gift from Northrop Grumman.

References

Davide Abati, Angelo Porrello, Simone Calderara, and Rita Cucchiara. 2019. Latent space autoregression for novelty detection. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 481–490. Computer Vision Foundation / IEEE.

Hessam Bagherinezhad, Hannaneh Hajishirzi, Yejin Choi, and Ali Farhadi. 2016. Are elephants bigger than butterflies? reasoning about sizes of objects. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 3449–3456. AAAI Press.

V. Barnett and T. Lewis. 1994. *Outliers in Statistical Data*. Wiley Series in Probability and Statistics. Wiley.

Shane Bergsma, Dekang Lin, and Randy Goebel. 2008. Discriminative learning of selectional preference from unlabeled text. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 59–68, Honolulu, Hawaii.

Yonatan Bisk, Rowan Zellers, Ronan LeBras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: reasoning about physical commonsense in natural language. 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 7432–7439. AAAI Press.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal.

Raghavendra Chalapathy and Sanjay Chawla. 2019. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*.

Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. 2018. Anomaly detection using one-class neural networks. *arXiv preprint arXiv:1802.06360*.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. 2015. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*.

Stephen Clark and David Weir. 2001. Class-based probability estimation using a semantic hierarchy. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark.

Tirthankar Dasgupta and Lipika Dey. 2016. Automatic scoring for innovativeness of textual ideas. In *AAAI Workshop: Knowledge Extraction from Text*.

Pradeep Dasigi and Eduard Hovy. 2014. Modeling newswire events using neural networks for anomaly detection. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1414–1422, Dublin, Ireland.

Joe Davison, Joshua Feldman, and Alexander Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1173–1178, Hong Kong, China.

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

Yihe Dong, Samuel B. Hopkins, and Jerry Li. 2019. Quantum entropy scoring for fast robust mean estimation and improved outlier detection. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 6065–6075.

Sarah M. Erfani, Mahsa Baktashmotlagh, Masud Moshtaghi, Vinh Nguyen, Christopher Leckie, James Bailey, and Kotagiri Ramamohanarao. 2017. From shared subspaces to shared landmarks: A robust multi-source classification approach. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 1854–1860. AAAI Press.

Sarah M Erfani, Sutharshan Rajasegarar, Shanika Karunasekera, and Christopher Leckie. 2016. High-dimensional and largescale anomaly detection using a linear one-class svm with deep learning. *Pattern Recognition*, page 121–134.

Katrin Erk and Sebastian Padó. 2010. Exemplar-based models for word meaning in context. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 92–97, Uppsala, Sweden.

Nausheen Fatma, Manoj Kumar Chinnakotla, and Manish Shrivastava. 2017. The unusual suspects: Deep learning based mining of interesting entity trivia from knowledge graphs. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 1107–1113. AAAI Press.

Geli Fei and Bing Liu. 2016. Breaking the closed world assumption in text classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 506–514, San Diego, California.

Geli Fei, Shuai Wang, and Bing Liu. 2016. Learning cumulatively to become more knowledgeable. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1565–1574. ACM.

Christiane Fellbaum. 2010. Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243.

M. Fey and J. E. Lenssen. 2019. Fast graph representation learning with pytorch geometric. *ArXiv*, abs/1903.02428.

Maxwell Forbes and Yejin Choi. 2017. Verb physics: Relative physical knowledge of actions and objects. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 266–276, Vancouver, Canada.

Maxwell Forbes, Ari Holtzman, and Yejin Choi. 2019. Do neural language representations learn physical commonsense? In *CogSci*.

Michael Gamon, Arjun Mukherjee, and Patrick Pantel. 2014. Predicting interesting things in text. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1477–1488, Dublin, Ireland.

Debasis Ganguly, Johannes Leveling, and Gareth Jones. 2014. Automatic prediction of aesthetics and interestingness of text passages. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 905–916, Dublin, Ireland.

Tirthankar Ghosal, Vignesh Edithal, Asif Ekbal, Pushpak Bhattacharyya, George Tsatsaronis, and Srinivasa Satya Sameer Kumar Chivukula. 2018. Novelty goes deep. a deep neural solution to document level novelty detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2802–2813, Santa Fe, New Mexico, USA.

Zhijiang Guo, Yan Zhang, and Wei Lu. 2019. Attention guided graph convolutional networks for relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 241–251, Florence, Italy.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, pages 1735–1780.

Wenpeng Hu, Mengyu Wang, Qi Qin, Jinwen Ma, and Bing Liu. 2020. HRN: A holistic approach to one class learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Binxuan Huang and Kathleen Carley. 2019. Syntax-aware aspect level sentiment classification with graph attention networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5469–5477, Hong Kong, China.

Seongung Jo, Heung-Seon Oh, Sanghun Im, and Seonho Kim. 2020. Cnn-based novelty detection with effectively incorporating document-level information. *KIPS Transactions on Computer and Communication Systems*, 9(10):231–238.

Frank Keller and Mirella Lapata. 2003. Using the web to obtain frequencies for unseen bigrams. *Computational Linguistics*, 29(3):459–484.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar.

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*.

Flip Korn, Xuezhi Wang, You Wu, and Cong Yu. 2019. Automatically generating interesting facts from wikipedia tables. In *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*, pages 349–361. ACM.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, pages 32–73.

Jingun Kwon, Hidetaka Kamigaito, Young-In Song, and Manabu Okumura. 2020. Hierarchical trivia fact extraction from Wikipedia articles. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4825–4834, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Xiaoyan Li and W Bruce Croft. 2005. Novelty detection based on sentence level patterns. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 744–751.

Shiyu Liang, Yixuan Li, and R. Srikant. 2018. Enhancing the reliability of out-of-distribution image detection in neural networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Ting-En Lin and Hua Xu. 2019. Deep unknown intent detection with margin loss. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5491–5496, Florence, Italy.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pages 413–422.

Nianzu Ma, Sahisnu Mazumder, Hao Wang, and Bing Liu. 2020. Entity-aware dependency-based deep graph attention network for comparative preference classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5782–5788, Online.

Kavi Mahesh and Pallavi Karanth. Smart-aleck: An interestingness algorithm for large semantic datasets. *algorithms*, 2:3.

Larry M Manevitz and Malik Yousef. 2001. One-class SVMs for document classification. *Journal of machine Learning research*, pages 139–154.

Matthew Merzbacher. 2002. Automatic generation of trivia questions. In *International Symposium on Methodologies for Intelligent Systems*, pages 123–130.

Tomas 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. European Language Resources Association (ELRA).

Dipannyta Nandi and Rohini Basak. 2020. A quest to detect novelty using deep neural nets. In *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–7. IEEE.

Duc Tam Nguyen, Zhongyu Lou, Michael Klar, and Thomas Brox. 2019. Anomaly detection with multiple-hypotheses predictions. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 4800–4809. PMLR.

Kazuya Niina and Kazutaka Shimada. 2018. Trivia score and ranking estimation using support vector regression and RankNet. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation*, Hong Kong.

Diarmuid Ó Séaghdha. 2010. Latent variable models of selectional preference. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 435–444, Uppsala, Sweden.

Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. 2021. Deep learning for anomaly detection: A review. *ACM Computing Surveys (CSUR)*, 54(2):1–38.

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)*, pages 1532–1543, Doha, Qatar.

Pramuditha Perera, Ramesh Nallapati, and Bing Xiang. 2019. OCGAN: one-class novelty detection using gans with constrained latent representations. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 2898–2906. Computer Vision Foundation / IEEE.

Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2015. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 2641–2649. IEEE Computer Society.

Ian Porada, Kaheer Suleman, and Jackie Chi Kit Cheung. 2019. Can a gorilla ride a camel? learning semantic plausibility from text. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 123–129, Hong Kong, China.

Amir Pouran Ben Veyseh, Nasim Nouri, Franck Dernoncourt, Quan Hung Tran, Dejing Dou, and Thien Huu Nguyen. 2020. Improving aspect-based sentiment analysis with gated graph convolutional networks and syntax-based regulation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4543–4548, Online.

Abhay Prakash, Manoj Kumar Chinnakotla, Dhaval Patel, and Puneet Garg. 2015. Did you know? - mining interesting trivia for entities from wikipedia. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 3164–3170. AAAI Press.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, page 9.

Philip Resnik. 1996. Selectional constraints: An information-theoretic model and its computational realization. *Cognition*, pages 127–159.

Alan Ritter, Mausam, and Oren Etzioni. 2010. A Latent Dirichlet Allocation method for selectional preferences. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 424–434, Uppsala, Sweden.

Liyun Ru, Le Zhao, Min Zhang, and Shaoping Ma. 2004. Improved feature selection and redundance computing-thuir at trec 2004 novelty track. In *TREC*.

Lukas Ruff, Nico Görnitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Robert A. Vandermeulen, Alexander Binder, Emmanuel Müller, and Marius Kloft. 2018. Deep one-class classification. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4390–4399. PMLR.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019. ATOMIC: an atlas of machine commonsense for if-then reasoning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3027–3035. AAAI Press.

Patrick Schlachter, Yiwen Liao, and Bin Yang. 2019. Deep one-class classification using intra-class splitting. In *2019 IEEE Data Science Workshop (DSW)*, pages 100–104.

B. Schölkopf, John C. Platt, J. Shawe-Taylor, Alex Smola, and R. C. Williamson. 2001. Estimating the support of a high-dimensional distribution. *Neural Computation*, pages 1443–1471.

Lei Shu, Hu Xu, and Bing Liu. 2017. DOC: Deep open classification of text documents. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2911–2916, Copenhagen, Denmark.

Lei Shu, Hu Xu, and Bing Liu. 2018. Unseen class discovery in open-world classification. *arXiv preprint arXiv:1801.05609*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. 2020. Csi: Novelty detection via contrastive learning on distributionally shifted instances. *Advances in Neural Information Processing Systems*, 33:11839–11852.

David M.J. Tax and Robert P.W. Duin. 2004. Support vector data description. *Machine Learning*, pages 45–66.

Ottokar Tilk, Vera Demberg, Asad Sayeed, Dietrich Klakow, and Stefan Thater. 2016. Event participant modelling with neural networks. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 171–182, Austin, Texas.

Trieu H Trinh and Quoc V Le. 2018. A simple method for commonsense reasoning. *arXiv preprint arXiv:1806.02847*.

Trieu H. Trinh and Quoc V. Le. 2019. Do language models have common sense?

David Tsurel, Dan Pelleg, Ido Guy, and Dafna Shahaf. 2017. Fun facts: Automatic trivia fact extraction from wikipedia. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017*, pages 345–354. ACM.

Tim Van de Cruys. 2009. A non-negative tensor factorization model for selectional preference induction. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 83–90, Athens, Greece.

Tim Van de Cruys. 2014. A neural network approach to selectional preference acquisition. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 26–35, Doha, Qatar.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Cunxiang Wang, Shuailong Liang, Yili Jin, Yilong Wang, Xiaodan Zhu, and Yue Zhang. 2020a. SemEval-2020 task 4: Commonsense validation and explanation. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 307–321, Barcelona (online). International Committee for Computational Linguistics.

Jingjing Wang, Sun Sun, and Yaoliang Yu. 2019. Multivariate triangular quantile maps for novelty detection. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5061–5072.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020b. Relational graph attention network for aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3229–3238, Online.

Su Wang, Greg Durrett, and Katrin Erk. 2018. Modeling semantic plausibility by injecting world knowledge. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 303–308, New Orleans, Louisiana.

Su Wang, Stephen Roller, and Katrin Erk. 2017. Distributional modeling on a diet: One-shot word learning from text only. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 204–213, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Zhibiao Wu and Martha Palmer. 1994. Verb semantics and lexical selection. In *32nd Annual Meeting of the Association for Computational Linguistics*, pages 133–138, Las Cruces, New Mexico, USA.

Y. Xiao and G. Zhou. 2020. Syntactic edge-enhanced graph convolutional networks for aspect-level sentiment classification with interactive attention. *IEEE Access*, pages 157068–157080.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2019. Open-world learning and application to product classification. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 3413–3419. ACM.

Chong You, Daniel P. Robinson, and René Vidal. 2017. Provable self-representation based outlier detection in a union of subspaces. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 4323–4332. IEEE Computer Society.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Brussels, Belgium.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy.

Min Zhang, Ruihua Song, Chuan Lin, Shaoping Ma, Zhe Jiang, Yijiang Jin, Yiqun Liu, Le Zhao, and S Ma. 2003. Expansion-based technologies in finding relevant and new information: Thu trec 2002: Novelty track experiments. *Nist special publication sp*, pages 586–590.

Sheng Zhang, Rachel Rudinger, Kevin Duh, and Benjamin Van Durme. 2017. Ordinal common-sense inference. *Transactions of the Association for Computational Linguistics*, 5:379–395.

Yi Zhang and Flora S Tsai. 2009. Combining named entities and tags for novel sentence detection. In *Proceedings of the WSDM'09 Workshop on Exploiting Semantic Annotations in Information Retrieval*, pages 30–34.

Panpan Zheng, Shuhan Yuan, Xintao Wu, Jun Li, and Aidong Lu. 2019. One-class adversarial nets for fraud detection. In *AAAI*, pages 1286–1293.

Yinhe Zheng, Guanyi Chen, and Minlie Huang. 2020. Out-of-domain detection for natural language understanding in dialog systems. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, pages 1198–1209.

A Dataset Statistics

A.1 Training data size for each verb

We selected 20 verbs in our data pool with sufficient scene descriptions containing each of these verbs so that we have enough data to learn commonsense knowledge. The list of 20 verbs and the size (in parentheses) of scene descriptions that contain these verbs are as follows: build (2644), carry (9920), climb(2001), cook (2232), cut (7103), drink(2050), drive (6913), eat (15822), fly (17049), hit (6316), jump (8947), kick (1759), look (31863), pull (6194), push (1901), ride (30244), swim (1760), throw (5299), travel (4410), walk (38254). There are totally 202,681 scene descriptions in our training dataset.

A.2 Dataset Split Detailed Statistics

Training Data. There are a total of 202,681 normal descriptions in the training data. The statistics of token numbers in descriptions are as follows: the average token number is 11.24, the maximum token number is 75, the minimum token number is 2. The standard deviation is 4.21.

Test Data. There are a total of 2000 normal descriptions and 2000 novel descriptions in the test data. The statistics of token numbers are as follows: the average token number is 11.10, the maximum token number is 68, the minimum token number is 4, the standard deviation is 3.8.

B Knowledge Based Contrastive Data Generator Details

In the novel scene detection task, we focus on three novelty aspects: (1) what actions an entity can perform, (2) what actions an entity can be applied, and (3) how several entities interact with each other. In the interactions between entities and verbs, verbs are the core of these interactions. Thus, we only replace entities that are syntactically related to a verb to create pseudo-novel description.

Extraction of candidate entities. The candidate entities are those syntactically related to a verb. If a sentence contains only a single verb, this sentence describes a single event and all nouns (noun phrases) are syntactically related to this verb. If a sentence contains multiple verbs, candidate entities for a target verb are the nouns (noun phrases) that are closer to the target verb along the dependency parse graph. In this multi-verb case, we create multiple training instances, one for each verb as the target verb. We use a simple rule-based extraction technique based on dependency parsing path and Part-of-Speech (POS) tagging to extract the relevant entities for each target verb. The nouns or noun phrases, one hop away to the target verb along the dependency parse graph are the candidate entities.

C General Hypernym Synsets

The 24 general synsets are: entity.n.01, abstraction.n.06, physical-entity.n.01, psychological-feature.n.01, causal-agent.n.01, object.n.01, group.n.01, thing.n.12, measure.n.02, matter.n.03, process.n.06, relation.n.01, attribute.n.02, communication.n.02, solid.n.01, part.n.01, part.n.02, part.n.03, state.n.02, solid.n.03, artifact.n.01, instrumentality.n.03, abstraction.n.06, whole.n.02.

D Graph Attention Network (GAT)

Graph Attention Network (GAT) (Velickovic et al., 2018) fuses the graph-structured information and node features within the model. Its masked self-attention layers allow a node to attend to neighborhood features, and to learn different attentions/weights to different nodes in the neighbors.

$$\boldsymbol{h}_i^{out} = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j\in\mathcal{N}_i} a_{ij}^k \boldsymbol{W}^k \boldsymbol{x}_j\right) \tag{2}$$

$$\alpha_{ij}^k = \frac{exp(f((\boldsymbol{a}^k)^T[\boldsymbol{W}^k\boldsymbol{x}_i \parallel \boldsymbol{W}^k\boldsymbol{x}_j]))}{\sum_{x\in\mathcal{N}_i} exp(f((\boldsymbol{a}^k)^T[\boldsymbol{W}^k\boldsymbol{x}_i \parallel \boldsymbol{W}^k\boldsymbol{x}_x]))}$$

The node features fed into a GAT layer are $\boldsymbol{X}=[\boldsymbol{x}_1, \boldsymbol{x}_2, ...\boldsymbol{x}_i, ...\boldsymbol{x}_n]$, $\boldsymbol{x}_i \in \mathbb{R}^F$, $\boldsymbol{X} \in \mathbb{R}^{n\times F}$, where n is the number of nodes, F is the feature size of each node. Specifically, in our context, each word corresponds to a node and F is the size of word embedding. In equation (2), node i attends over its 1-hop neighbors $j \in \mathcal{N}_i$. $\Vert_{k=1}^{K}$ means the concatenation of K multi-head attention outputs. $\boldsymbol{h}_i^{out} \in \mathbb{R}^{F'}$ is the output of node i at the current layer. α_{ij}^k is the k-th attention between node i and j. $\parallel$ is the concatenation operation. $\boldsymbol{W}^k \in \mathbb{R}^{\frac{F'}{K}\times F}$ is linear transformation. $\boldsymbol{\alpha} \in \mathbb{R}^{\frac{2F'}{K}}$ is the weight vector, and $f(\cdot)$ is a LeakyReLU non-linearity function.

Overall, the input-output for a single GAT layer is summarized as $\boldsymbol{H}^{out} = GAT(\boldsymbol{X}, \boldsymbol{A}; \Theta)$. The input is $\boldsymbol{X} \in \mathbb{R}^{n\times F}$ and the output is $\boldsymbol{H}^{out} \in \mathbb{R}^{n\times F'}$, where n is the number of nodes, F is the node feature size, F' is GAT hidden size, and $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is the adjacency matrix of the graph.

E GAT-MA Model Implementation Details

We employ Stanford Neural Network Dependency Parser (Chen and Manning, 2014) to convert each scene description into dependency parse graph. In our experiments, two pretrained embedding are used: GloVe[9] (Pennington et al., 2014) embedding and BERT[10] (Devlin et al., 2019) embedding. To produce BERT embedding, the input of BERT is formatted by adding "[CLS]" before and "[SEP]" after the tokens of the description. This input is tokenized by BERT tokenizer into word pieces. The output of the pretrained BERT model embedding is a sequence of vectors, each of size 768. Each output vector corresponds to one word piece token. BERT tokenizer tokenizes some words into word pieces (sub-word tokens), such as "tokenizer" is tokenized as word pieces "token" and "##izer". We take the average of the word pieces embedding of the original word to obtain embedding of this word. Note that, we use BERT embedding as the static input feature for GAT-MA. The model does not fine-tune BERT.

We empirically set GAT-MA hyper-parameters as follows: hidden state size as 300D; BERT embeddings mapped into 300D using a linear layer. 6 attention heads used for the GAT layers. The mini-batch size is set as 256 and learning rate is set as 5e-5. We use larger batch size to make training process faster. We apply 0.1 embedding dropout (Srivastava et al., 2014) and 0.1 attention dropout. We apply l_2 regularization with term $\lambda = 10^{-4}$. Adam (Kingma and Ba, 2015) optimizer is used for training. The model is trained with 5 epochs. Each epoch takes around 200 minutes to run.

The implementation of this model is based on PyTorch Geometric(PyG) (Fey and Lenssen, 2019) and NVIDIA GPU GTX 1080 Ti.

F Baseline Models Implementation Details

For all the baselines, we do experiments using various embeddings, such as GloVe, BERT and InferSent embeddings. We report the best results for comparison.

[9]We use glove.840B.300d in our experiments

[10]We use the BERT model "bert-base-uncased" as text encoder. We expect that using larger transformer embedding leads to better results. But due to our limitation of computational resources, we only did experiments based on this base BERT model.

F.1 Language Model Based Novelty Detector

N-gram. N-gram is a classic language model that can assign probabilities to a sequence of words. We do experiments with the choice of $N \in \{1, 2, 3, 4, 5\}$. Among them, $N = 1$ gives the best result.

LSTM. GloVe embedding is used to train the LSTM model on our training dataset. The embedding size is 300. The hidden layer size of LSTM is 300. The number of stacked LSTM layers is 2. The dropout applied to the LSTM layer during training is 0.5. The initial learning rate is 50. The learning rate lr is annealed by equation $lr = lr/4.0$ if no improvement has been seen in the validation dataset.

BERT. We fine-tune the pretrained BERT with our training data following the default setting of the original BERT paper. Because BERT is a masked language model, the probability of word i in a list of tokens is obtained by mask this word and calculate the probability this word appearing in the current context. The context of word i is the tokens on both left and right side of this word in the description.

GPT-2. We fine-tuned GPT-2[11] on our training data following the default setting. Then we use the trained model to calculate the sentence probability. The word probability is calculated by checking its probability appearing in its context. The context of word i in the sentence is the token on the left side of this word.

F.2 General One-Class Classifiers

Most of the general one-class classifiers work on image data. We change the components in the baseline models into structures of text encoder to make them applicable to text data. We have tried 4 methods to produce sentence embedding as follows. (a) **GloVe-AVG**: taking average of all words' GloVe embeddings as the sentence embedding, (b) **BERT-CLS**: feeding a description into BERT and using the first token [CLS] as the sentence embedding, (c) **BERT-AVG** feeding a description into BERT and taking the average of the sequence output embedding as the sentence embedding and (d) **InferSent**: feeding a description into the pre-trained sentence embedding extractor InferSent (Conneau et al., 2017; Bowman et al., 2015) to produce the sentence embedding. There are two versions of

[11]gpt2: 12-layer, 768-hidden, 12-heads, 117M parameters; OpenAI GPT-2 English model.

InferSent pretrained model. **InferSent-1** is trained using GloVe embedding. **InferSent-2** is trained using fastText (Mikolov et al., 2018) embedding.

OCSVM. The parameter setting of OCSVM are as follows: we use “poly" kernel; gamma as “scale", nu value as 0.1. For other parameters, we use the default setting in the scikit-learn implementation[12]. OCSVM gets the best result using GloVe-AVG sentence embedding.

iForest. The parameter setting of iForest are as follows: we use 100 base estimators in ensemble. For the amount of contamination of dataset, we set it as 0.0 because there is no novel scene description in our training dataset. For other parameter settings, we follow the default settings in the scikit-learn implementation[13]. iForest gets the best result using InferSent-1 embedding.

VAE. We use the text encoder structure in Convolutional Neural Networks (CNN) for Sentence Classification (Kim, 2014) to implement a VAE that can take text data as input. For all hyper-parameters, we follow the settings from the original work. Among the 4 methods of converting descriptions into sentence embeddings, BERT-CLS gets the best result.

DSVDD. The LeNet implementation is used as our baseline model. The latent dimension of the autoencoder as well as the final fully connected layer of the model is changed to a dimension of 96 to better accommodate the size of our description embeddings. For all other parameters, we use the default settings from the original work and implementations. Among the 4 methods of converting descriptions into sentence embeddings, BERT-CLS gets the best result.

ICS. We use the default settings from the original work and implementations. Among the 4 methods of converting descriptions into sentence embeddings, BERT-CLS gets the best result.

OCGAN. To make OCGAN work better for text data, we change the depth of the generator and discriminator from 3 layers to 2 layers, the noise factor of training data from 0.02 to 0.05, and the weight of the reconstruction loss from 500 to 600. For other hyper-parameters, we follow the settings in the original work. Among the 4 methods of converting descriptions into sentence embeddings, BERT-CLS gets the best result.

HRN. We followe the default setting of the original paper. We run HRN 100 epochs, with batch size 100. The learning rate is set as 0.0003. The structure of Multilayer Perceptron (MLP) is [768-300]-[300-100]-[100-1]. HRN gets the best result using BERT-CLS sentence embedding.

[12] sklearn.svm.OneClassSVM

[13] sklearn.ensemble.IsolationForest