Unveiling the Type of Relationship Between Autonomous Systems Using Deep Learning

Tal Shapira
School of Electrical Engineering
Tel-Aviv University
talshapira1@mail.tau.ac.il

Yuval Shavitt
School of Electrical Engineering
Tel-Aviv University
shavitt@eng.tau.ac.il

***Abstract*—The ToR inference problem had been widely investigated in the last two decades, mostly using heuristic algorithms. In this problem, we attempt to reveal the economic relationships between ASes, data with applications in network routing management and routing security.**

In this paper, we introduce a novel approach for ToR classification, which is based on embedding the AS numbers (ASN) in high dimensional space using neural networks. Similar to natural language processing (NLP) models, the embedding represents latent characteristics of the ASN and its interactions on the Internet. The embedding coordinates of each AS are represented by a vector; thus, we call our method BGP2VEC. In order to solve the supervised learning problem presented, we use these vectors as an input to an artificial neural network and achieve a state of the art accuracy of 95.2% for ToR classification.

***Index Terms*—Deep Learning, Internet, BGP, AS relationships, AS embedding**

I. Introduction

The Internet consists of thousands of Autonomous Systems (ASes), each AS operated by an administrative domain such as an Internet Service Provider (ISP), a business enterprise, or a University. Each autonomous system is assigned a globally unique number, the Autonomous System Number (ASN), and advertises (announce) one or more IP address prefixes (APs) using the Border Gateway Protocol (BGP). BGP routing's update messages list the entire AS path to reach an AP. For each AP, BGP allows each AS to choose which routes to accept (import policy), how to select the best route, and whether to announce it (export policy).

The commercial agreements between two connected ASes are broadly classified into three types of relationship (ToR) [1]: 1) Provider-to-customer (P2C) - the customer AS pays the provider AS for transit traffic from and to the rest of the Internet, 2) Peer-to-peer (P2P) - two ASes freely exchange traffic between themselves and their customers, but do not exchange traffic from or to their providers or other peers, and 3) Siblings (S2S) - two ASes that belong to the same administrative domain. Gao [1] defined concatenation rules for AS links in a route that model the way ASes usually configure their BGP, it is called Valley Free (VF) since once a route descend from a provider to a customer it cannot ascend again. An interesting observation from the VF model is that connectivity does not imply reachability, and the shortest path in the (undirected) AS graph may not be usable due to the BGP VF constraints.

ToR information allows us to infer the possible routes selected by BGP, e.g., in case of a link failure [2]. It can also be used to identify malicious fiddling with the routing system, known as IP hijack attack [3], [4]. However, ToR information is mostly not public, and thus there is a long line of research to infer it [1], [5], [6], [7], [8]. Most of these solutions are heuristic algorithms based on publicly available BGP announcement databases [9], [10]. An inherent problem in these algorithms is their use of heuristics, causing unbounded errors that are spread over all inferred relationships.

Over the past few years, advances in deep learning [11] have driven tremendous progress in many fields; one of them is Natural Language Processing (NLP). We build on the excellent results achieved for NLP tasks (Word2Vec [12]), were word adjacency in sentences is used to map words to a large dimensional space. Instead, we use adjacency of ASNs in BGP announcements to embed ASes in a large (we selected 32) dimensional space and attach to each AS a vector of its coordinates in this space. Based on the ASN embedding, we apply artificial neural networks for the ToR classification problems.

Our approach achieved excellent results: we classify AS ToRs with an accuracy of 95.2%. As far as we know, we are the first to solve this problem using deep learning methods. We should also mention that the embedding also allows us to classify ASNs, which is outside the scope of this paper.

The rest of the paper continues as follows. After describing related work in Sec. II, we describe the datasets in Sec. III. In Sec. IV we describe our two-stage method, which is based on ASN embedding, i.e., *BGP2VEC*, and applying artificial neural network for the classification task. Sec. V presents our experiments and their results with comparison to previous results. Finally, the last section concludes the paper.

II. Related Work

There are many works that focused on solving the problem of inferring Autonomous Systems (ASes) type of relationships (ToR), most of them proposed heuristic algorithms based on extracting information from BGP announcements or based on generating AS level Routes from traceroutes. We will focus here on works which are based on BGP route information, and

978-1-7281-4973-8/20/$31.00

disregard works that leveraged other information to improve ToR inference, such as usage of BGP communities or IXP route servers [13].

Gao [1] was the first to study the AS relationships inference problem. She presented heuristic algorithms that infer AS ToRs from BGP routing announcements based on the fact that a provider's AS graph degree is usually larger than its customers, and that peers have about the same degree. The algorithm locally identifies the top provider for each path and classifies edges (ToRs) following the valley-free nature of routing paths.

Subramanian *et al.* [5] introduced the ToR maximization problem, which is to label all the edges in an undirected AS graph, in order to maximize the number of valley-free paths in a set of BGP routes. Their algorithm exploits the structure of partial views of the AS graph, as seen from different locations. For each location, it calculates the rank of each AS using a reverse-pruning algorithm, and infer the ToR between two ASes by comparing their vectors of ranks; if the ranks are similar, the algorithm classifies the link as P2P, otherwise as C2P.

Xia and Gao [14] used the BGP Community Attribute, the AS-SET object, and the routing policies in the IRR Databases to infer AS relationships. However, their approach only obtains partial AS relationships (14% of total AS pairs on Oct. 2003). They showed that both GAO [1] and SARK [5] inferred poorly P2P relationships.

Battista *et al.* [15] proved that the ToR optimization problem [1], [5] is NP-complete, and reduced the problem to the ToR-D problem that allows a small number of invalid paths. They reduced the ToR-D problem to 2SAT and introduced a heuristic algorithm for determining the ToRs. Cohen and Raz [6] defined the Acyclic Type of Relationship (AToR) problem that attempts both to minimize the number of invalid paths and keep the directed graph acyclic. They introduced a heuristic algorithm to solve the K-AToR problem.

Dimitropoulos *et al.* [7] used the IRR [10] to infer S2S relationships and then introduced a more realistic problem formulation that accepts that AS paths do not always exhibit a hierarchical pattern to infer P2C and P2P relationships. Their algorithm introduced a metric called reachability, sorted all ASes by their reachability, and grouped ASes with the same value into levels. They correctly inferred 96.5% C2P, 82.8% P2P, and 90.3% S2S relationships.

Shavitt *et al.* [8] were motivated to reduce the usage of heuristics. They proposed a near-deterministic algorithm for solving the ToR inference problem (ND-ToR), that uses the Internet's core (a sub-graph of top-level ASes), which was constructed in three different ways: the Greedy Max Clique (GMC) core [16], the k-Core which is based on the k-shell decomposition [17], and the CAIDA Peers Core (CP) which is the largest connected component of a P2P graph (provided by CAIDA [18])- that contains some of the largest tier-1 ASes. They inferred the rest of the links using a three-phase algorithm based on the valley-free rule, and the k-shell index [17] of the adjacent ASes. Their algorithm succeeded to infer over 95% of approximately 58,000 ToRs based on AS-level paths collected from RouteViews [9] and DIMES [19].

Luckie *et al.* [20] introduced the AS-Rank algorithm for inferring C2P and P2P links using BGP data. Their work relies on three assumptions: 1) there is a clique of large transit providers at the top of the hierarchy, 2) most customers enter into a transit agreement to be globally reachable, and 3) cycles of C2P links should not exist for routing to converge. Based on these assumptions, they introduced a new algorithm for inferring the customer cone of an AS, which is the set of ASes that the AS can reach using P2C links, and achieved state of the art results.

In order to overcome the inference barriers for hard cases, such as non-valley-free routing, limited visibility, and non-conventional peering practices, Jin *et al.* [21] identified key interconnection features and developed a probabilistic algorithm (ProbLink), and showed that their algorithm achieved an error rate that is better than AS-Rank over their validation set. However, they use additional information, such as sibling relationships, BGP communities, and IXP information.

As mentioned in [14], [7], [8], [20], [22], the existing heuristic algorithms rely on assumptions such as the presence of valley-free paths, the existence of a peering clique of ASes at the top of the hierarchy and more. **This highly motivated our work for introducing a deep learning based approach that relies only on the characteristic of the data (BGP routes).** As far as we know, this is the first time deep learning is used for this problem. As we will show, our deep learning method produced significantly better results than previous rule-based and heuristic algorithms.

Table I: Number of labeled ToRs in the dataset.

CAIDA AS ToRs serial-2		
P2P	P2C	C2P
608,486	118,405	118,405

III. The Datasets

In our experiments, we use data that was collected in March 2018. We use two types of datasets:

1) **RouteViews's BGP announcements (RV)** [9] - contains BGP path announcements collected from 19 route collectors. The dataset consists of approximately 3,600,000 BGP paths, 62,525 AS vertices, and approximately 113,400 undirected links. We use this unlabeled dataset for the first stage of our approach (i.e., ASN embedding).
2) **CAIDA AS Relationships Dataset** [23] - provides two datasets: serial-1, which contains 343,952 P2P pairs and 118,405 P2C/C2P pairs, that were inferred from BGP paths using AS-Rank [20], and serial-2, which contains additional 264,534 P2P pairs that were inferred from BGP communities attributes using the method described in [13]. The total number of samples is displayed in Table I. We use this labeled dataset as labels for training our neural network and various supervised learning algorithms based on ASN embedding, and as a benchmark for

comparison with previous works. Note that the CAIDA dataset does not contain siblings.

IV. Method

Our method works as follows: first, using a shallow neural network, we map each ASN to an embedded vector. Then, for the ToR classification task, we activate Artificial Neural Network (ANN) that receives the vectors from the previous stage. In this section, we will introduce in details both stages.

A. ASN Embedding

Applications of neural networks have expanded significantly in recent years [11]. One of them is embeddings, a method that is used to represent discrete variables as continuous vectors. This technique is broadly used in the field of Natural Language Processing (NLP), also known as word embeddings [12], [24], which helps machine learning algorithms to achieve better performance by grouping similar words. Embeddings are important for input to a neural network, as it is trained to work on vectors of real numbers. Moreover, the method produces vectors such that similar words have close vectors, where similarity is defined in terms of both syntax and semantics.

As mentioned in Sec. I, in the first stage, we produce a 32-dimensional continuous vector representation for each ASN. As in the training process of word embedding in NLP, i.e., Word2Vec [12], we train our network over a large corpus of AS paths (described in Sec. III), which are equivalent to the sentences in NLP tasks. We apply a similar skip-gram model as introduced in [12], such that for each ASN in a certain AS-path we predict ASNs within a certain range before and after the current ASN, (as shown in Fig. 1). As a result, the model learns to characterize an ASN by its context, i.e., neighboring ASNs.

Our model contains an input layer of size 62,525 (the number of distinct ASes in our dataset, denoted by V), one fully connected (FC) hidden layer whose size is the embedding size N (using a grid search method, we found that an embedding of size 32 achieves best results, see Sec. V-B), and an output layer whose size is determined by the window size. The hidden layer weight matrix is of a size VxN, such that each ASN in the corpus corresponds to an N-features vector; these are the ASN-vectors that are learned by the model. The output layer is the *softmax layer* [25], whose size is V for each desired output.

We choose to apply a window of size 2 (see Figure 1), which is the maximum distance between the input ASN and a predicted ASN (the output) within an AS path, which results with an output layer with a maximum size equals to 4x62,525. In order to improve the representation, we use negative sampling [12], to distinguish the target ASN from the noise distribution using 5 negative samples for each target ASN. We build and run our network using the *Gensim* [26] library.

The training procedure is done by feeding the network with the ASN pairs; the input is a one-hot vector representing the input ASN and the training outputs, which are also one-hot vectors representing the output ASNs (the context ASNs). Then applying gradient descent learning [27] (also known as back-propagation) to adjust the weights of the network in order to maximize the log probability of any context word given the input word.

Table II: The architecture of our artificial neural network.

ToR Classification	
Layer Type	Input/Output Size
Embeddings:	Input: 2, 1 Output: 2, 32
Conv1D:	Output: 2, 32
MaxPool:	Output: 2, 16
Conv1D:	Output: 32, 16
MaxPool:	Output: 16, 16
Fully Connected:	Output: 100
Softmax:	Output: 3

B. ANN Architecture

For the ToRs classification problem, we choose to use a simple ANN (see Table II) comprised of seven layers, not counting the input. A sequence of ASNs is fed into the first layer of the network, which is an embedding layer. Each ASN is embedded into a 32-dimensional vector based on the first stage. The next layer is a 1-dimensional convolutional layer [28] (labeled as ConV1D) followed by ReLU [29] activation function with 32 filters of length 3 and a total number of 3,104 trainable parameters (3072 weights and 32 bias parameters). The next layer is a max-pooling layer with 32 feature maps of size 2, where each unit in each feature map outputs the maximum value of 2 neurons in the corresponding feature map in ConV1D. The next layer is a second ConV1 layer (with 224 trainable parameters) followed by a second max-pooling layer. The next layer is a fully-connected layer with 100 neurons and a ReLU activation function (with 25700 trainable parameters). Finally, our output layer is the softmax layer with 3 outputs, one for each class. The last layer contains 303 trainable parameters.

The training of the neural network is done by optimizing the *categorical cross entropy* [30] cost function, which is a measure of the difference between the softmax layer output and a one-hot encoding vector of the same size, representing the correct label of the sample. For the optimization process, we use the *Adam* [31] optimizer, which is an extension to the stochastic gradient descent algorithm. We use the default hyper-parameters as provided in Kingma *et al.* [31] and set our batch size to 64.

We build and run our networks using the *Keras* [32] library with *Tensorflow* [33] as its back-end. We use 80% of the samples as a training set and 20% of the samples as a test set. We split each dataset such that the ratio between the quantities of the classes remains the same in both the training set and the test set, while there is no ToR in the training set in which its inverse appears in the test set. We run our network for 40

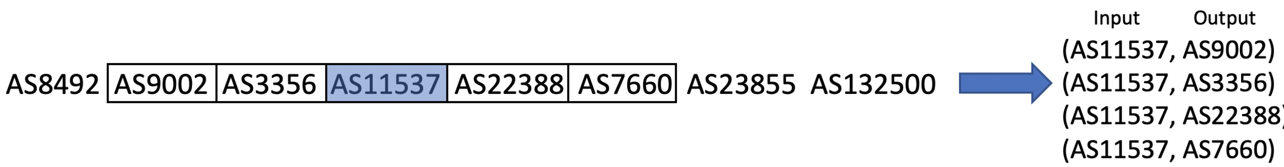

Figure 1: An example for generating input/output ASNs for the training process of BGP2VEC using a window size of 2.

epochs of the training set. During the test time, our network classifies a ToR in an average time of 0.1 milliseconds.

V. Experiments and Results

In this section, we report our experimental results by a comparison of our ToR classification to best known previous results. Moreover, we will show that our method had found many mistakes in the CAIDA dataset, which emphasizes the strength of our results.

Due to a lack of space, we omit experimental results that demonstrate the strength of the ASN embedding. We can attest that the ASN embedding captures many latent characteristics of the ASNs. For example, the closest ASN to AS3356 is its sibling AS3549, and the next 3 nearest neighbors are other tier-1 providers: Telia, Verizon, and KPN. All the 5 nearest neighbors of IUCC (The Israeli universities' network, AS378) with high cosine similarity scores are also Educational/Research ASes from Europe and the Middle East with a small degree. None of these ASes are connected directly.

Table III: A comparison of the ToR classification accuracy and recalls.

Algorithm	Accuracy	P2P Rc.	C2P Rc.	P2C Rc.
CAIDA AS Relationships Dataset (serial-1)				
GAO	38.4%	1.0%	99.1%	85.7%
SARK	81.9%	16.1%	95.4%	95.2%
NDToR CP-CORE	89.0%	99.9%	70.8%	71.2%
NDToR TS-CORE	56.2%	54.7%	33.3%	82.3%
NDToR Kmax-CORE	84.6%	12.3%	99.5%	99.4%
RUAN	78.6%	2.3%	97.8%	97.7%
***BGP2VEC* - NN**	**94.2%**	89.0%	93.1%	98.5%
***BGP2VEC* - LR**	81.6%	93.7%	51.1%	49.0%
***BGP2VEC* - SVM**	76.5%	89.1%	58.6%	57.7%
***BGP2VEC* - KNN**	92.6%	94.0%	90.0%	90.1%
***BGP2VEC* - D-KNN**	92.1%	94.0%	89.6%	89.%
***BGP2VEC* - KMeans**	61.6%	79.8%	26.6%	43.6%
CAIDA AS Relationships Dataset (serial-2)				
BGP2VEC - NN	**95.2%**	98.0%	88.4%	87.6%

A. Evaluation Criteria

We use the **accuracy** criteria to evaluate our model performance, which is defined as the proportion of examples for which the model produces the correct output of all predictions made. A formal definition of the accuracy for multiclass classification is

$$Accuracy = \frac{\sum_{i \in classes} TP_i}{\sum_{i \in classes}(TP_i + FP_i)},$$

where TP_i and FP_i are the true positive and the false positive of the class i, respectively. Moreover, for each class we also calcultae the the *recall*, defined by $Rc = \frac{TP}{TP+FN}$ (where FN is the false negative).

B. Hyper-parameters Optimization

Figure 2 shows the results of a grid search over different ASN embedding sizes (V) and window sizes, in order to optimize these parameters to achieve the best accuracy for ToRs classification. We performed a grid search for window sizes of 1, 2, and 3; and for embedding sizes in powers of 2 ranging from 2 to 512. In each experiment, we used the same architecture as depicted in Sec. IV, with only one modification, which is the output of the embedding layer.

The results show that embedding sizes greater than 8 and different window sizes give similar results for the ToRs classification problem. Since increasing the embedding size increases dramatically the number of parameters in the neural network, we select an embedding size of 32 and a window of 2.

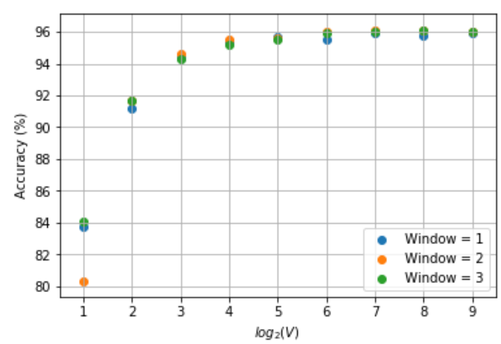

Figure 2: Grid search results for the ToRs classification's accuracy as a function of the embedding size (V) and the window size of the BGP2Vec.

C. ToR Classification Results

Table III compares *BGP2VEC* results to other previously suggested algorithms based on our CAIDA AS Relationship serial-1 dataset, which contains AS relationships inferred from BGP using the method described in [20]. In our comparison, we focused only on methods that are based on AS-level paths, thus do not include the PTE [14] algorithm. Shavitt *et al.* [8] observed a problem when executing the heuristic phases for inferring P2P (and S2S) relationships of the AToR [6] and BPP [15] algorithms. Thus, we compare only the P2C and C2P assignments of both algorithms. In our experiments, the AToR algorithms succeeded to accurately infer 93.7% and 98.1% of the C2P ToRs and P2C ToRs, respectively, while BPP infer only 83.7% and 68.2% of these ToRs.

We implemented the following algorithms for the comparison: 1) the first ToR inference algorithm which was proposed by Gao [1] (referred to as GAO), 2) the SARK algorithm that was presented by Subramanian *et al.* [5], 3) the near deterministic AS ToRs inference algorithm [8] (i.e., ND-ToR) using three different cores; CP-Core and kmax-Core as described in Sec II, and a new core that is based on the training set (referred as TS-Core), which contains approximately 90,000 randomly chosen labeled ToRs (which are 80% of our ToR dataset), and 4) the most recent algorithm (RUAN) that was introduced by Ruan and Susan Varghese [22]. We implement this algorithm based on 19 Tier-1 ASes according to the clique at the top of the hierarchy of the CAIDA relationships dataset [23].

The presented algorithms can not take advantage of the labeled training set without a significant change. Even when we tried to take advantage of the training set and constructed a new core for the ND-ToR (NDToR TS-CORE), which consists of the training set edges with their labeled ToRs, the method achieves poor results, relative to the other cores.

A comparison of the ToR classification results is presented in Table III. Our method achieves the best performance, with an accuracy of **95.8%**, 5.2% higher than the second-best algorithm (NDToR CP-Core).

In the second part of Table III, we show our *BGP2VEC* algorithm results over the CAIDA AS Relationships serial-2 dataset, which adds about 300k links inferred from BGP communities using the method described in [13]. Despite the imbalanced dataset, our method succeeds in achieving even better results compared to the serial-1 dataset with an accuracy of 95.2%. Previous works, which were described before, achieved similar results for both datasets.

We manually explored some misclassified test results. For the 30 misclassified ToRs with the highest softmax scores (all above 0.999), 12 were correct (i.e., incorrect labels in the dataset), 12 were indeed misclassifications, 3 were siblings, and 3 were unclear. Namely, of the classifiable ToRs, half of our mistakes were actually correct.

For example, the ToR [AS5009, AS6939] is labeled by CAIDA as P2P, while our network predicts it correctly as C2P. By examining AS5009 routing, one can easily infer that AS 6939 is its main transit provider since almost all of AS5009 traffic is traversing through AS6936 (Hurricane Electric). Since March 2018, CAIDA corrected this ToR. Another example is the ToR [AS52614, AS267221], which is labeled by CAIDA as P2C, while our network predicts it correctly as C2P. According to AS52614's IRR AS267221 is its provider, which is also clear by the fact that many of its route traverses AS267221. We could not find any route from AS267221 that traverse AS52614.

Finally, we examined 25 ToRs that do not exist in our dataset. Typically, these are ASes from the edges of the Internet, where our database probably has many missing ToRs or misclassified ToRs (ASes from Brazil and Nepal together comprised of 38% of the ToR's endpoint). For this challenging group, we got 18 correct classifications (most with softmax scores above 0.95), 4 errors (only 1 with high softmax score), and 3 siblings.

Overall, when the softmax score is sufficiently high, our classification seems to work very well. The threshold seems to be in the range [0.85, 0.9], but it requires more manual data tagging to pin the exact threshold.

An interesting finding in our small manual experiments is the large percentage of sibling ToRs. This may hints that many of our misclassifications are attributed to siblings.

D. Testing BGP2VEC Embedding Performances Using Different Machine Learning Algorithms

In order to emphasize the strength of the BGP2VEC embedding, we apply 'classic' supervised and unsupervised machine learning algorithms for the ToRs classification problem. Table III summarizes our results using a 64-dimensions embedding-vector for each Tor, by concatenating together two 64-dimensions embedding-vectors, one for each AS.

We tested two basic supervised learning algorithms, Logistic Regression (denoted as LR) and Support Vector Machine (SVM), which achieve accuracies of 81.6% and 76.5%, respectively. As can be seen in Table III, both methods struggle to classify C2P/P2C ToRs, which is mainly due to the imbalanced dataset (for example, by applying weighted loss to balance the imbalances in the data, we achieve similar recalls for both P2P, C2P, and P2c ToRs).

In order to understand the strength of the AS-pairs similarity, for each AS-pair we generate a distance-embedding, i.e., we subtract the embedding of the second AS from the embedding of the first AS and get a 32-dimensions distance-vector. Then we apply the K-Nearest Neighbours (KNN) algorithm over the distance-vectors (denoted as D-KNN in Table III), which choose the ToR which achieves a majority vote among the 5 nearest neighbors. The D-KNN achieves an accuracy of 92.1%, which is the best accuracy achieved over the symmetric dataset. We also apply a KNN algorithm using the concatenated 64-vectors and achieve an accuracy of 92.6%. Figure 3 displays the number of neighbors with the same ToR, based on the KNN algorithm with K=5 over the symmetric test set for; P2P, C2P, P2C and combined. As can be seen, 83.0% of the 5-neighbors ToRs are identical, which means that the distance-vectors that achieved by the BGP2VEC embedding characterize well the ToRs.

Last, we apply the K-Means unsupervised-learning algorithm. Then we determine the class of each cluster by applying a majority vote. We achieve a best accuracy of 61.6% with K=10. This result shows that although similar ToRs are located close to each other (as can be concluded by the KNN results), they are not spatially arranged in clusters.

In summary, our deep learning method achieves state of the art results in a relatively short evaluation time for ToRs classification. Moreover, we show that ToRs can be inferred using simple machine learning algorithms based on BGP2VEC embeddings.

VI. Concluding Remarks

In this paper, we introduce a novel approach for numerical characterization of ASes using deep learning methods and

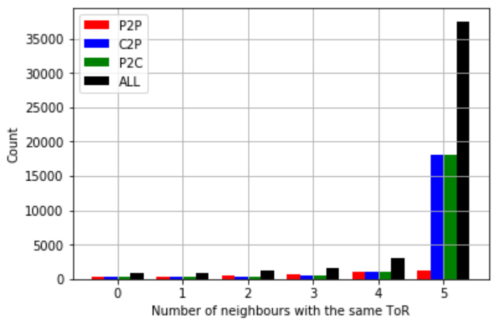

Figure 3: Number of neighbours with the same ToR based on KNN with K=5 over the symmetric test set for; P2P, C2P, P2C and combined.

apply it to achieve state of the art results for AS Type of Relationships (ToRs) classification. Our solution consists of two stages: learning dense representations of ASNs from BGP routes (*BGP2VEC*), and applying ANN using the ASN embeddings as inputs for the classification. As far as we know, we are the first to employs deep learning for this problem.

We had tested our algorithm on the CAIDA AS Relationship dataset and found it to perform very well, with 95.2% accuracy. Manual inspection showed that of the 4.8% of the misclassifications, almost half were correct and are due to errors in the CAIDA dataset. By comparing our method with previous works, our method achieves the best performance, 5.2% higher than the second-best algorithm. Moreover, our method found mistakes in the labeled dataset, which demonstrates the strength of our results.

This work is the first in this direction, and we plan to use it as a building block for other problems, such as detecting hijacked routes from BGP announcements. We also plan to explore the ASN embeddings further to reveal the latent characteristics of ASes.

Acknowledgment

This research was funded in part by a grant on cyber research from the Israeli PMO, and by the Blavatnik Interdisciplinary Cyber Research Center at Tel Aviv University.

References

[1] L. Gao, "On inferring autonomous system relationships in the internet," *IEEE/ACM Transactions on Networking*, vol. 9, no. 6, pp. 733–745, Dec 2001.
[2] D. Dolev, S. Jamin, O. Mokryn, and Y. Shavitt, "Internet resiliency to attacks and failures under BGP policy routing," *Computer Networks*, vol. 50, Nov. 2006.
[3] C. C. Demchak and Y. Shavitt, "China's maxim - leave no access point unexploited: The hidden story of china telecom's BGP hijacking," *Military Cyber Affairs*, vol. 3, Oct. 2018.
[4] P. Sermpezis, V. Kotronis, A. Dainotti, and X. Dimitropoulos, "A survey among network operators on BGP prefix hijacking," *ACM SIGCOMM Computer Communication Review (CCR)*, vol. 48, no. 1, pp. 64–69, Jan 2018.
[5] L. Subramanian, S. Agarwal, J. Rexford, and R. H. Katz, "Characterizing the internet hierarchy from multiple vantage points," in *INFOCOM'02*, vol. 2, June 2002, pp. 618–627 vol.2.
[6] R. Cohen and D. Raz, "Acyclic type of relationships between autonomous systems," in *IEEE INFOCOM 2007*, 2007, pp. 1334–1342.
[7] X. Dimitropoulos, D. Krioukov, M. Fomenkov, B. Huffaker, Y. Hyun, k. claffy, and G. Riley, "AS relationships: Inference and validation," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 1, pp. 29–40, Jan. 2007.
[8] U. Weinsberg, Y. Shavitt, and E. Shir, "Near-deterministic inference of AS relationships," in *ConTel 2009*, Zagreb, Croatia, Jun. 2009.
[9] U. of Oregon Advanced Network Technology Center, "Route views project," http://www.routeviews.org/, 2018.
[10] I. R. Registry, http://www.irr.net/, 2018.
[11] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.
[13] V. Giotsas, S. Zhou, M. Luckie *et al.*, "Inferring multilateral peering," in *The ninth ACM conference on Emerging networking experiments and technologies*, 2013, pp. 247–258.
[14] J. Xia and L. Gao, "On the evaluation of AS relationship inferences," in *IEEE GLOBECOM'04*, vol. 3, Nov. 2004.
[15] G. D. Battista, M. Patrignani, and M. Pizzonia, "Computing the types of the relationships between autonomous systems," in *IEEE INFOCOM 2003.*, vol. 1, March 2003, pp. 156–165.
[16] S. L. Tauro, C. Palmer, G. Siganos, and M. Faloutsos, "A simple conceptual model for the internet topology," in *Global Telecommunications Conference (GLOBECOM'01)*, vol. 3. IEEE, 2001, pp. 1667–1671.
[17] S. Carmi, S. Havlin, S. Kirkpatrick, Y. Shavitt, and E. Shir, "A model of internet topology using k-shell decomposition," *Proceedings of the National Academy of Sciences (PNAS)*, vol. 104, no. 27, pp. 11 150–11 154, 2007.
[18] C. Rank, http://as-rank.caida.org/, 07 2018.
[19] Y. Shavitt and E. Shir, "DIMES: Let the internet measure itself," *SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 5, pp. 71–74, Oct. 2005.
[20] M. Luckie, B. Huffaker, A. Dhamdhere, V. Giotsas *et al.*, "AS relationships, customer cones, and validation," in *Internet Measurement Conference*. ACM, 2013, pp. 243–256.
[21] Y. Jin, C. Scott, A. Dhamdhere, V. Giotsas, A. Krishnamurthy, and S. Shenker, "Stable and practical AS relationship inference with ProbLink," in *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*, Boston, MA, USA, Feb. 2019, pp. 581–598.
[22] L. Ruan and J. Susan Varghese, "Computing observed autonomous system relationships in the internet," *Computer Science Technical Reports*, no. 367, 2014.
[23] T. C. R. Dataset, http://www.caida.org/data/active/as-relationships/, 2018.
[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *CoRR*, vol. abs/1301.3781, 2013. [Online]. Available: http://arxiv.org/abs/1301.3781
[25] Y. T. Zhou, R. Chellappa, A. Vaid, and B. K. Jenkins, "Image restoration using a neural network," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, no. 7, pp. 1141–1151, Jul 1988.
[26] R. Řehůřek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora," in *the LREC 2010 Workshop on New Challenges for NLP Frameworks*, Valletta, Malta, May 2010, pp. 45–50.
[27] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[28] Y. L. Cun, O. Matan, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jacket, and H. S. Baird, "Handwritten zip code recognition with multilayer networks," in *10th International Conference on Pattern Recognition*, vol. 2, Jun. 1990, pp. 35–40.
[29] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. USA: Omnipress, 2010, pp. 807–814.
[30] D. Campbell, R. A. Dunne, and N. A. Campbell, "On the pairing of the softmax activation and cross–entropy penalty functions and the derivation of the softmax activation function," in *8th Australian Conference on Neural Networks*, Melbourne, Australia, 1997, pp. 181–185.
[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.
[32] F. Chollet *et al.*, "Keras," https://github.com/fchollet/keras, 2015.
[33] M. A. et al., "Tensorflow," https://www.tensorflow.org/, 2015.