Towards a systematic multi-modal representation learning for network data

Zied Ben Houidi, Raphael Azorin, Massimo Gallo, Alessandro Finamore, Dario Rossi

Huawei Technologies Co. Ltd

ABSTRACT

Learning the right representations from complex input data is the key ability of successful machine learning (ML) models. The latter are often tailored to a specific data modality. For example, recurrent neural networks (RNNs) were designed having the processing of sequential data in mind, while convolutional neural networks (CNNs) were designed to exploit spatial correlation naturally present in images. Unlike computer vision (CV) and natural language processing (NLP), each of which targets a single well-defined modality, network ML problems often have a mixture of data modalities as input. Yet, instead of exploiting such abundance, practitioners tend to rely on sub-features thereof, reducing the problem on single modality for the sake of simplicity.

In this paper, we advocate for exploiting all the modalities naturally present in network data. As a first step, we observe that network data systematically exhibits a mixture of *quantities* (e.g., measurements), and *entities* (e.g., IP addresses, names, etc.). Whereas the former are generally well exploited, the latter are often underused or poorly represented (e.g., with one-hot encoding). We propose to systematically leverage state of the art embedding techniques to learn entity representations, whenever significant sequences of such entities are historically observed. Through two diverse use-cases, we show that such entity encoding can benefit and naturally augment classic quantity-based features.

1 Introduction

Deep learning's success is mainly due to its ability to *learn good representations* from complex unstructured data. Such ability is a fundamental aspect of intelligent agents, both artificial and biological. The representation learning ubiquity is perhaps best witnessed by the striking similarities between features learned by artificial neural networks and biological brains. Two representative examples are *visual* and *spatial* representations. Decades after the discovery of simple and complex cells by 1983 Nobel prize winners Hubel and Wiesel [17, 16], it was found that deep artificial neural networks learn strikingly similar simple-to-complex representations [21, 5]. The same applies for the 2014 Nobel prize winning discovery of Place and Grid cells [11, 8], which are neurons that encode internal representations of places and space. Similar representations were discovered in artificial agents that learn how to navigate [1, 7]. Advances in Natural Language Processing (NLP) similarly corroborate the importance of learning good representations, by (i) pre-training neural networks on large unlabeled datasets with self-supervised tasks, followed by (ii) per-task fine tuning with few labels – which is behind the success of GPT-3 in NLP few-shot learning [3]. A similar process also proved crucial for few-shot image classification – where learning a good representation or embedding, followed by training a simple linear-classifier on top of it outperformed state of the art few-shot methods [27].

Casting these observations to networking, to fully exploit machine learning potential, it seems necessary to put more focus on *representation learning of network data*. This is all the more important, given the abundance of unlabeled data generated and collected by networks. Such appeal is however immediately moderated by the complexity of network data, namely its *multi-modality*. Indeed, the most prominent advances in machine learning have been obtained on classic single modalities. From the perspective of input data, NLP takes sequences of categorical variables as input and CV takes as input pixel values stored in fixed-size matrices representing images. Additionally, within the same language, words have a coherent meaning across contexts and corpora. The same applies to visual features which are "universal" across domains to some extent. This is far from being the case in network data which is way more heterogeneous (including multi-variate timeseries, flow and system logs, devices configuration, topologies, routing events, etc.) and where identifiers may have a more "local" significance.

Lacking a universal network data representation, machine learning has been applied to network problems in a rather opportunistic way, either focusing on a specific modality, or handcrafting input features by mean of expert knowledge. On the opposite side, each classic modality in mainstream machine learning tasks has its own research community. For example, CV heavily relies on variations of CNNs (AlexNet[20] , ResNet[12] or MobileNet[15]) to handle images tasks (e.g., classification, segmentation). Modern NLP models instead take as input vector representations of words and sub-words, pre-trained using some word embedding technique (e.g., word2vec [24] or ELMo [25]) on large corpora of raw text. Sequence to sequence models (first long-short term memory[13] then transformers [28]) are then often applied on such sequence of vector representations to solve a language task (e.g., classification, translation).

As such, a legitimate yet challenging question emerges: *"what is the representation learning strategy that is best fit for the various modalities of network data?"*. It is exactly to answer this question that we call for research arms in this paper. We believe that in order to take full advantage of emerging machine learning techniques, the networking community must rethink its "retina" (i.e., the input data format) and "visual cortex" (i.e., the representation learning strategy used to extract knowledge from the input). Even assuming that for each modality there exists a different learning strategy, it is unlikely that one needs to invent a new machine learning discipline for each of them. Alternatively, and more realistically, one could map each of the existing network modalities to the best-fit existing representation learning technique – which is the starting point of this paper.

Taking a first principled step beyond uni-modality, we remark the existence of a natural dichotomy in network data, where we identify two network data types: *quantities* which are measured features such as numbers of packets, bytes, etc.; *entities*[1] which instead range from the named objects that relate to these measurements (e.g., source IP, user id) to various attributes or events' names (e.g., "interface down"). As we argue about the similarity between sequences of co-occurring network traffic entities and sequences of words in natural language, we postulate that language model pre-training is the best tool to learn a representation for such data. Indeed, similarly to natural language, the order and context in which network entities co-appear in network logs is often not arbitrary, and hence patterns could be learned from it. For example, in NLP, recent word embedding techniques [24, 25] proved remarkably powerful in extracting deep semantic relationships between words from their co-occurence in raw corpora. Accordingly, we propose to systematically leverage language model pre-training to learn vector representations, also known as *embeddings*, whenever (i) significant sequences of such entities are *historically* observed and (ii) these entities are *consistently named* across time and space.

Throughout the paper, we refer to this network data dicotomy as entity-quantity *bimodality*, that we systematically explore as a first principled step towards network data multimodality. In particular, we advocate for the need to use language model pre-training, such as word embeddings, to learn rich entities representations. The latter can then be simply concatenated with quantities (or their auto-encoded representation[19]), before performing a learning task. We illustrate our proposal in two diverse toy cases: (i) *clickstream identification*, where entities are sequences of domain names that carry a semantic meaning, as well as (ii) *terminal movement prediction*, where entities are access points identifiers that are not expected to have any semantic.

In the remainder of the paper, we abstract our bimodal approach in Sec. 2 and apply it to our illustrative use cases in Sec. 3. Finally, we show supporting examples from the literature in Sec. 4 and discuss future opportunities in Sec. 5.

2 A bimodal representation for network data

As a first step, we narrow the scope to a family of network data which we believe is representative for a significant portion of data collected in networking. We then provide some background on word embeddings which are our chosen language model pre-training method. We illustrate why and under which conditions, they are suitable to deal with what we call sequences of entities. We conclude by illustrating a generic prototype of the bimodal pipeline.

2.1 Entities and quantities in network data

While producing a thorough taxonomy of all network data types is a challenging and useful target, it is outside the scope of this paper. Instead, as a first step, we simply notice the difference between two main families of data types, for which a unified representation learning strategy could be devised.

As argued earlier, most of network data concern a mixture of entities and quantities that evolves over time. Quantities represent telemetry derived by various measurement apparatus, while entities are abstract objects often related to them. The latter are described by names that are assigned by humans (e.g., trouble tickets, error messages found later in the logs, IP addresses, domain names, host identifiers in general), hence carrying a semantic meaning. We further argue that *sequences of entities* carry precious information encoded in the non-arbitrary order in which the elements appear in the sequence (i.e., the "context" in which the entities co-occur, with one another and with the quantities). We advocate that such sequences must be systematically leveraged, and that NLP word embedding and self-supervised pre-training are the appropriate representation learning techniques.

Of course, we acknowledge that not all network data is sequential or measured over time (a typical example is static topology snapshots). However such data still often pertains to entities (e.g., node names or identifiers) for which sequence data is abundant in parallel sources (e.g., routing information), in which case our proposed representation learning guidelines are still applicable to some extent.

2.2 Word and character embeddings

Oversimplifying, the closest problem in ML communities is the learning from sequential data in NLP - words are nothing more than sequences of entities that follow each other. To perform a machine learning task, words must be first transformed into a numerical representation. This can be naïvely done using integer or one-hot encoding. The last years however have witnessed the emergence of tools that became almost standard for NLP tasks. Instead of operating on raw one-hot encoded vectors, modern NLP models, either build or take as input word vector representations

[1] In machine learning frequently identified as categorical variables.

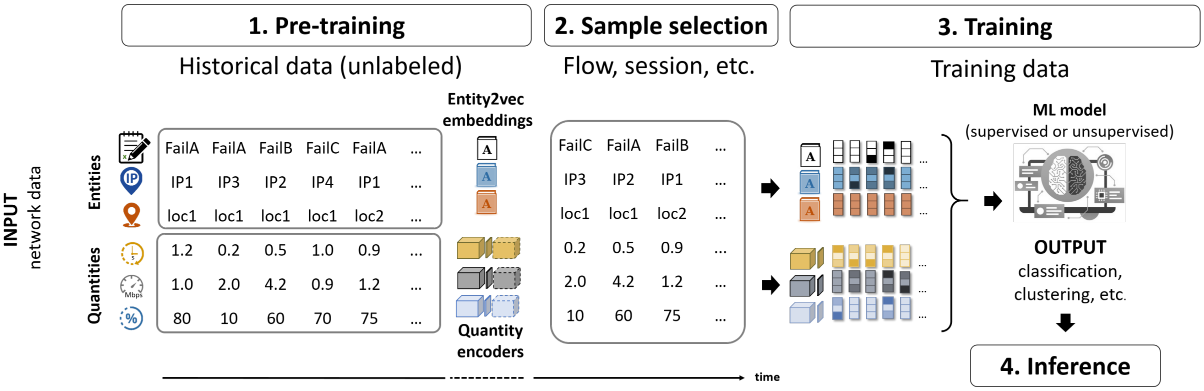


Figure 1: Generic bimodal pipeline.

obtained through *self-supervised pre-trained models* created from large text corpora. One famous word *embedding* technique, which we use in this paper as an example for its simplicity, is word2vec [24]. It transforms each word into a high dimensional vector, hence "embedding" it into an hyperspace. In practice, word embeddings are sometimes complemented with subword or character level embeddings meaning that sub-words/characters themselves have vector representations [18, 4]. This allows to account for out of vocabulary words, misspellings, etc.

2.2.1 *Building vector representations with word2vec*

With word2vec, in a nutshell, a simple neural network with one hidden layer (whose dimensions are those of the embedding vectors to be learned) is trained to predict a target word from its surrounding **context**. Word2vec thus does not need expensive or human-made labels, but rather cheaply builds labels to supervise the training from the sequence data itself, thus it is self-supervised. First, all words are encoded using one-hot encoding, resulting in a one-hot vector of the size of the vocabulary in which each position represents one word. The neural network is then trained on large amounts of sequences of words from which the **(context, word)** pairs are extracted for training. At the end of the training, the neural network used to predict target words is no longer used, only the weights are. In particular, for each position in the one-hot encoding, the learned weights are used to form the vector representing the corresponding word. Two main parameters hence influence the learned representations: the size of the embedding layer, and the size of the context window with which labels are built.

2.2.2 *Emerging properties*

Although trained to "simply" predict the next word in a sequence, the vector representations learned by word embeddings exhibit interesting properties. The most mediatized example is the ability to extract semantic relationships by doing simple arithmetic operations on vectors (e.g., King - Men + Woman = Queen). Another popular example is that vector representations of different languages exhibit strikingly similar structures, such that it is possible to use a few anchors to align the vector representations of two languages and find that words with similar meaning fall in the same "positions" in each language vector [23]. This same observation opened the way later to self-supervised language translation using only single-language corpora [22].

2.2.3 *Conditions to apply language model pre-training*

When applied on natural languages, techniques like word2vec and language model pre-training in general do not impose particular conditions that the language to model must satisfy. However, we believe that when moving away from natural language and generalizing to any arbitrary sequence of named entities, *at least* two conditions must be satisfied. The foremost is the *(i) consistency of naming*. Like words in natural language, network entities are expected to always keep the same meaning[2]. Moreover, we require *(ii) stability* of the corpora: while this is implied in natural language as adding a new word to the vocabulary is an unfrequent event, observing a new entity in sequence network data is rather frequent.

Drawing the proper conclusion from these above conditions, we can infer that sequences of entities containing, e.g., non-consistently anonymized IP addresses, are not suitable for entity embedding. Instead, entities that are named consistently and *relatively* stable over time are good candidates.

2.3 A bimodal pipeline

We sketch a prototype implementation of a bimodal pipeline, consisting of four steps namely *Pre-training*, *Sample selection*, *Training* and *Inference*, as shown in Figure 1.

Pre-training on historical sequences. Similarly to pre-training in classic ML, our first phase consists in leveraging huge amounts of unlabeled data to learn relevant representations from the different data types. As exemplified in the leftmost part of Figure 1, the pipeline takes sequences of various unlabeled networking data as input. Quantities and en-

[2] However some exceptions may exist, e.g., "set" has different meanings depending on the context. As such, contextual word embeddings like ELMo [25] have been devised that solve this problem.

tities are then fed to the most suitable representation learning pipelines, e.g., auto-encoder for quantities and word embeddings for entities.

Input sample selection Once embeddings are trained, the next step is to define the input samples for downstream tasks. By *input sample*, we mean the individual subject that the next ML task will take as input. Unlike some classic ML tasks whose subject samples are often clear (either a matrix of pixel values for image classification or a sequence of strings for translation or sentiment analysis), network-related ML tasks may have a variety of subjects. For example, if the goal is to classify IP addresses (respectively flows) as either malicious or benign, then the input sample should be a feature vector representation of the IP address (respectively the flow). Alternatively, the input sample could be the first N packets of a flow, or a sequence of flows, etc. Once the input sample is decided, its fixed-size vector representation is created by combining (i) the corresponding quantities (or their representations in case of auto-encoding) and (ii) the learned entity representations. For the sake of simplicity, in the reminder of this paper entities' embbedings and quantities are combined by simple concatenation.

Training downstream ML tasks When pre-training and sample selection is done, training a downstream task is rather straightforward. For instance, in an unsupervised use-case, one can cluster the vector representations. Likewise, in a supervised classification use-case, one can associate labels to the vector embeddings to train the classifier. Notice that in this case, the more robust representation learned, the fewer the labeled samples required for training [27, 3]. In our clickstream toy case, we use a relatively small synthetically built labeled dataset leveraging page visits of top 1000 Alexa ranking websites.

Inference and models updates The last step once models are learned, is to use them to perform inference. Classic ML challenges on how to keep the models up-to-date clearly apply here, but are out of scope in this paper.

3 Use cases

With the objective of demonstrating the validity of the proposed bimodal approach, this section illustrates the advantages of embedding network entities, using two illustrative use-cases **clickstream identification** and **mobile terminal movement prediction**. It is worth mentioning that to showcase our approach, we only focus on categorical data embeddings and use an *as-is* network quantities representation (i.e., raw unprocessed data). Otherwise stated, we use word2vec pre-training for entities and leave quantities as they are: while a better representation of quantities might exist, we leave it for future work. Compared to the practitioner approach to resolve to unimodal learning, these two use cases allow us to demonstrate the usefulness of categorical entities embedding – as their use can either improve or outpeform results gathered by quantitative data.

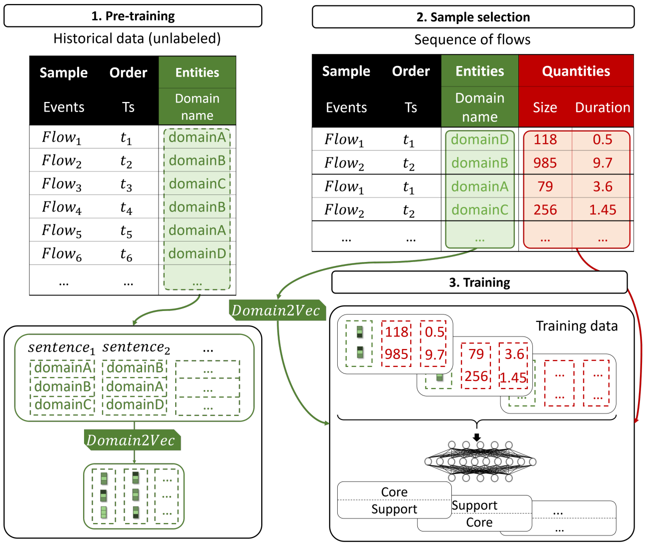


Figure 2: Bimodal pipeline for the clickstream use case.

3.1 Clickstream identification

In modern Web traffic, a single page corresponds to the download of tens of objects, retrieved from tens of different locations.[3] Since the advent of encryption, an ISP which collects flow logs of web traffic would only observe a series of entries with no clue of (i) which entry belongs to which page, nor (ii) which flow queries the domain of the main page i.e., *core domain*, and which flows query a necessary resource to render the page i.e., *support domains*. Such knowledge could be useful for example to estimate per-page Web quality of experience metrics from flow logs. Beyond the usefulness of the scenario itself, the two tasks above comes with a number of methodological challenges that we believe are well suited to illustrate the proposed bimodal representation learning scheme.

As shown in Figure 2, a network vantage point collects per-flow size, duration, measurements (quantities) and related domain names (entities). Following our guidelines, the first step is pre-training. In our case, we imagine sequences of entities as words in a language, thus we train a *domain2vec* model that is later used (at training and inference time) to embed domain names. As earlier mentioned, domain names can be embedded either with word or both word and character embeddings. With character embedding, words like `cdn`, `cdn21`, and `cdn22` will have similar embeddings even if they never co-occur in similar contexts.

Given the two tasks described above, an input sample in our case is a series of consecutive flows, corresponding to the simultaneous query of an unknown number of web pages. A first ML task for us aims to "disentangle" flows by associating each of them to its Web page. A subsequent task then is to classify flows as either core- or support-related. We relied on a Gated Recurrent Unit (GRU) model for such binary

[3] Respectively 70 and 50 in our top 1000 Alexa dataset.

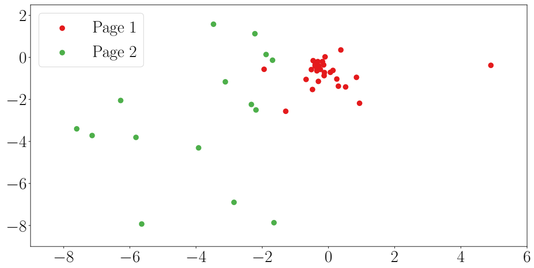


Figure 3: 2D PCA visualization of the domain embeddings of two different pages.

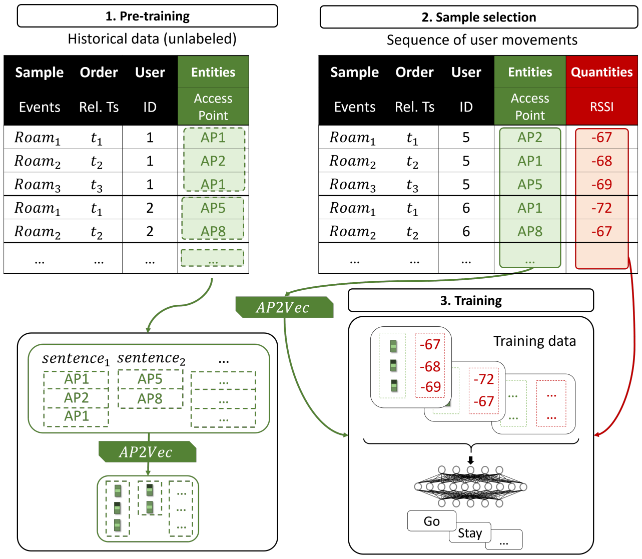
Approach	Precision			Recall		
	Q1	Q2	Q3	Q1	Q2	Q3
Naïve	100%	100%	100%	14%	16%	20%
Quantities	66%	75%	100%	50%	60%	75%
Word emb.	75%	100%	100%	33%	50%	66%
Word + Char emb.	80%	100%	100%	58%	77%	87%

Table 1: Comparison of different approaches to the click-stream use case.

task. We qualitatively show how our entity-based representation helps solving these tasks. For each sample, we compare different representations: quantity-only features (e.g., flow size and byte progression related metrics), and word embedding only, word and character embeddings.

For our evaluation, we consider a Web traffic dataset containing 20k domains retrieved by downloading 10 times each of the Web pages from top 1,000 Alexa ranking, while recording in parallel flow logs. The pre-training dataset is then constructed by synthetically generating 100k multisessions of 3 to 10 (median of 6) simultaneous page visits. Domain2vec is then trained using a context window of 200 flows that generate a vector representation with 200 dimensions. The supervised learning dataset on the other hand consists of 60k similarly synthesized multisessions. We select around 900 pages to build the training and validation sets (50k multisessions). We test on 10k multisessions containing composed by theremaining *unseen* 100 pages which queried more than 1.2k unseen support domains. All in all, training/validation and test sessions queried respectively 15k and 2k domains.

As a first qualitative illustration, Figure 3 plots a 2 component PCA representation of domain embeddings belonging to two web pages. Interestingly, without additional feature learning, the domain embeddings of different pages are already “disentangled”, i.e., flows of different pages cluster in different regions, thus hinting that entity-based embeddings extract useful features. More quantitatively, Table 1 presents the results of our classification model, focusing on precision and recall of the minor class, i.e., core domain. In addition to the earlier discussed representations, we show as a baseline the results of a naïve predictor which systematically tags the first domain as core (and hence correctly predicts that one but misses the others in the multisession). For each multisession, we compute a precision and a recall in predicting the core domains. We show the 3 quartiles across all multisessions. Despite the difficulty of the task, all models performed better than the naïve baseline. Surprisingly, entity-based encoding alone outperformed the quantity-based one. Less surprising, character embedding adds value compared to word embedding only. Note that here, one-hot encoding could not have been a viable solution because (let alone its prohibitive memory cost) the test set contains only unseen pages.



Sample Events	Order Rel. Ts	User ID	Entities Access Point
$Roam_1$	t_1	1	AP1
$Roam_2$	t_2	1	AP2
$Roam_3$	t_3	1	AP1
$Roam_1$	t_1	2	AP5
$Roam_2$	t_2	2	AP8
...	...	...	...

Sample Events	Order Rel. Ts	User ID	Entities Access Point	Quantities RSSI
$Roam_1$	t_1	5	AP2	-67
$Roam_2$	t_2	5	AP1	-68
$Roam_3$	t_3	5	AP5	-69
$Roam_1$	t_1	6	AP1	-72
$Roam_2$	t_2	6	AP8	-67
...	...	...	...	

Figure 4: Bimodal pipeline for the movement prediction use case.

3.2 Movement prediction in Wireless LANs

A Wireless LAN deployment typically involves several access points (AP) providing network connectivity to mobile terminals (e.g., cellphones, laptops). In this context, one problem is to predict early enough whether a terminal is going to move away from its access point, reconnecting to another one. This allows the network operator to proactively steer the terminal to roam before its signal actually degrades.

As depicted in Figure 4, the available network data are series of received signal strength indicators (RSSI) (quantities) and the set of APs traversed over time (entities). In other words, each terminal has a current associated AP and a signal strength towards it. According to the proposed bimodal pipeline, at pre-traing the RSSI and AP lists are grouped, this time by user ID. The sequences of per-user APs are then used to train an AP2vec embedding model using word2vec. Once the embedding model is trained, the downstream task is modeled with a 1D CNN using as input the concatenation of the RSSI series with the related AP embedding.

To evaluate the performance of the bimodal pipeline in this use case, we consider a dataset including 5 days of real network data with approximately 2k mobile terminals and 240k

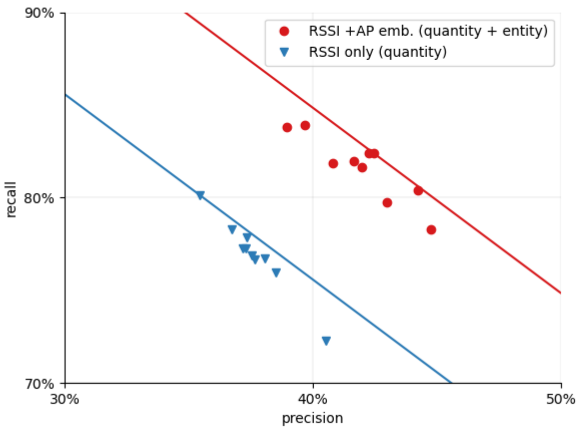


Figure 5: Comparison of movement prediction models.

movement events across 80 different AP. The AP2vec pretraining is executed on 10 AP long sequences and generate an embedding vector with 20 dimensions. The 1D CNN model is then trained with samples composed by the last 10 RSSIs, and the last AP vector representation. The supervised learning training dataset is composed by the first 3 days while the test one the remaining 2 days.

Figure 5 presents the precision-recall scatter plot obtained with the 1D CNN model when RSSI-only and RSSI+AP2vec embeddings are used as input. From the results it is clear that the bimodal network data representation help the ML task to reach a more accurate and stable movement prediction (as it can be observed in the RSSI plot over time, not reported here due to lack of space). It is worth mentioning that in this case, given the limited number of entities i.e., 80, one-hot encoding is also a viable solution as embedding technique. Despite finding the optimal embedding technique is out of the scope of this work, we report that surprisingly the sample composed by RSSI and AP2vec always lead to a slightly better model.

4 Related work

With the advent of machine learning, in the last few years the networking community started to explore different ways of representing networking data and in particular categorical ones. One of the first attempts in this direction is IP2Vec [26] which embeds network packets' source and destination IP addresses and ports with the objective of identifying IP addresses with similar behaviors. However, the embedding is limited to the 5-tuple itself and does not fully exploit the power of word2vec embedding when used with consecutive sequences of flows. Another important work that uses embedding in the context of Network data is DANTE [6] which encodes with a word2vec-like approach port sequences contacted by attackers with the goal of identifying malicious behaviors. Similarly to DANTE, Darkvec [9] uses word embedding to project potential attackers, identified by IP, and grouped by service, identified by ports, into a latent space with the goal of clustering senders with similar behaviour. Another interesting example that exploits the power of embeddings is [10] that is close to the clickstream usecase. Authors use browsing historical data to generate user profiles by means of representation learning techniques such as word2vec.

The alternative to finding a suitable representation for the networking data at hand is feature engineering. As previously discussed, feature engineering in the context of network data frequently ends up in using only a single modality, typically the quantity. For instance, in [2] authors study the trade-off between model accuracy and the computational cost of feature extraction at line rate. To do so, they developed Traffic refinery, a framework to select a proper data representation (i.e., through feature selection) that is both effective (i.e., achieve good accuracy) and feasible (i.e., can be deployed at line rate). nPrintML [14] takes an orthogonal approach to network data representation with respect to the one proposed in this paper, by encoding packets in a one-hot encoding format that is then used to feed classical ML/DL models. While capturing all features from network data, such approach is extremely costly and fails to identify relevant patterns.

5 Conclusions

This paper argues for the need of a systematic, unified and multi-modal representation learning for network data. As a first step, we propose a principled bimodal network data representation of *entities* and *quantities*, in which historical sequences of *entities* are systematically transformed into vector representations using word and sub-word embedding techniques. We show the effectiveness of such representation through two new toy examples, as well as referring to recent published examples from the literature. As systems and network data are rife with sequential events, we believe that the scope for potential applications of bimodal data representation learning scheme such as the one proposed here are broader than the illustrative toy cases. Relevant data not considred in this preliminary work include sequences of IP addresses, BGP advertisements or routing events, system logs, alarms, etc., and applications are likewise numerous.

Yet, network data is more complex than the co-occuring sequences of events and quantities illustrated in this paper. As such, we believe the introduced bimodal representation to be a useful conceptual framework – otherwise stated, bimodality is only the *starting point of the journey* towards modeling more general multi-modality network data. For instance, as entities often exhibit complex relationships that can be represented by time-evolving graphs, Graph Neural Networks (GNN) and graph embedding techniques seem to be another necessary piece in the quest toward multi-modality. Incorporating these pieces in the bigger puzzle of network data representation remains an interesting open research question for the networking community as a whole.

6 References

[1] A. Banino, C. Barry, B. Uria, C. Blundell, T. Lillicrap, P. Mirowski, A. Pritzel, M. J. Chadwick, T. Degris, J. Modayil, et al. Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705):429–433, 2018.

[2] F. Bronzino, P. Schmitt, S. Ayoubi, H. Kim, R. Teixeira, and N. Feamster. Traffic refinery: Cost-aware data representation for machine learning on network traffic. *Proc. ACM Meas. Anal. Comput. Syst.*, 5(3), dec 2021.

[3] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[4] X. Chen, L. Xu, Z. Liu, M. Sun, and H. Luan. Joint learning of character and word embeddings. In *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[5] R. M. Cichy, A. Khosla, D. Pantazis, A. Torralba, and A. Oliva. Comparison of deep neural networks to spatio-temporal cortical dynamics of human visual object recognition reveals hierarchical correspondence. *Scientific reports*, 6(1):1–13, 2016.

[6] D. Cohen, Y. Mirsky, M. Kamp, T. Martin, Y. Elovici, R. Puzis, and A. Shabtai. Dante: A framework for mining and monitoring darknet traffic. In *Computer Security – ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I*, page 88–109. Springer-Verlag, 2020.

[7] C. J. Cueva and X.-X. Wei. Emergence of grid-like representations by training recurrent neural networks to perform spatial localization. *arXiv preprint arXiv:1803.07770*, 2018.

[8] M. Franzius, H. Sprekeler, and L. Wiskott. Slowness and sparseness lead to place, head-direction, and spatial-view cells. *PLoS computational biology*, 3(8):e166, 2007.

[9] L. Gioacchini, L. Vassio, M. Mellia, I. Drago, Z. B. Houidi, and D. Rossi. Darkvec: automatic analysis of darknet traffic with word embeddings. In *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies*. ACM, 2021.

[10] R. Gonzalez, C. Soriente, J. M. Carrascosa, A. Garcia-Duran, C. Iordanou, and M. Niepert. User profiling by network observers. In *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies*. ACM, 2021.

[11] T. Hafting, M. Fyhn, S. Molden, M.-B. Moser, and E. I. Moser. Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801–806, 2005.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[13] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[14] J. Holland, P. Schmitt, N. Feamster, and P. Mittal. New directions in automated traffic analysis. *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, Nov 2021.

[15] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[16] D. H. Hubel and T. N. Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3):574, 1959.

[17] D. H. Hubel and T. N. Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology*, 195(1):215–243, 1968.

[18] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush. Character-aware neural language models. In *Thirtieth AAAI conference on artificial intelligence*, 2016.

[19] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR*, 2014.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[21] N. Kruger, P. Janssen, S. Kalkan, M. Lappe, A. Leonardis, J. Piater, A. J. Rodriguez-Sanchez, and L. Wiskott. Deep hierarchies in the primate visual cortex: What can we learn for computer vision? *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1847–1871, 2012.

[22] G. Lample, A. Conneau, L. Denoyer, and M. Ranzato. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*, 2017.

[23] T. Mikolov, Q. V. Le, and I. Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013.

[24] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[25] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. arxiv 2018. *arXiv preprint arXiv:1802.05365*, 12, 1802.

[26] M. Ring, A. Dallmann, D. Landes, and A. Hotho. Ip2vec: Learning similarities between ip addresses. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, page 657–666, 2017.

[27] Y. Tian, Y. Wang, D. Krishnan, J. B. Tenenbaum, and P. Isola. Rethinking few-shot image classification: a good embedding is all you need? In *European Conference on Computer Vision*, pages 266–282. Springer, 2020.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.