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
first.(mid.)last@huawei.com

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 sequential data in mind, while convolutional neural networks (CNNs) were designed to exploit spatial correlation 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 to 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 language models 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.

CCS CONCEPTS

• **Networks** → **Network performance analysis**; • **Computing methodologies** → **Learning latent representations**; **Knowledge representation and reasoning**;

HotNets '22, November 14–15, 2022, Austin, TX, USA

ACM ISBN 978-1-4503-9899-2/22/11…$15.00
https://doi.org/10.1145/3563766.3564108

KEYWORDS

Multimodal representation learning, Network data

ACM Reference Format:
Zied Ben Houidi, Raphael Azorin, Massimo Gallo, Alessandro Finamore, Dario Rossi. 2022. Towards a systematic multi-modal representation learning for network data. In *The 21st ACM Workshop on Hot Topics in Networks (HotNets '22), November 14–15, 2022, Austin, TX, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3563766.3564108

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, 18], it was found that artificial neural networks learn similar simple-to-complex representations [6, 22]. The same applies for the 2014 Nobel prize winning discovery of Place and Grid cells [9, 12], for which some neurons encode places and space representations. Similar representations were found in artificial agents that learn how to navigate [2, 8]. Advances in NLP also 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 few-shot learning [4]. A similar process proved crucial for few-shot image classification – where learning good representations or embeddings, followed by training simple linear-classifiers outperformed state-of-the-art few-shot methods [28].

Casting these observations to networking, to fully exploit ML potential, it seems necessary to put more focus on *representation learning for 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 ML 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. 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, topologies, routing events, etc.) and where identifiers may have a more "local" significance.

Lacking a universal network data representation, ML has been applied to network problems in a rather opportunistic way focusing on a specific modality, or handcrafting input features. On the opposite side, each classic modality in mainstream ML tasks has its own research community. For example, CV heavily relies on variations of CNNs (AlexNet[21] , ResNet[13] or MobileNet[16]) to handle images tasks (e.g., classification, segmentation). Modern NLP models instead take as input words and sub-words vector representations, pre-trained using word embedding technique e.g., word2vec [25] or ELMo [26], on large corpora of raw text. Sequence to sequence models (long-short term memory[14] and transformers [29]) 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 network data modalities?"*. 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 ML techniques, the networking community must rethink its "retina" i.e., input data format, and "visual cortex" i.e., 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 each of them require a new ML discipline. 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. and *entities* (or categorical in ML) 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, using appropriate language models [25, 26]. Accordingly, we propose to 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 dichotomy as entity-quantity *bimodality*, that we 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 representation [20]), before performing a ML task. To get a better understanding of the proposed approach we illustrate our proposal in two toy cases: (i) *clickstream identification*, where entities are sequences of domain names that carry a semantic meaning, and (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 NETWORK DATA REPRESENTATION

As a first step, we narrow the scope to a representative family of network data. We then provide some background on word embeddings used as language model pre-training method, illustrating why and under which conditions they are suitable to deal with network data. We conclude by presenting a generic design of a bimodal pipeline.

2.1 Entities and quantities in network data

While producing a thorough taxonomy of 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, network data include a mixture of entities and quantities evolving over time. Quantities represent telemetry derived by measurement apparatus, while entities are abstract objects often related to them. The latter are typically described by names assigned by humans e.g., trouble tickets, IP addresses, domain names, host identifiers, etc., 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". We advocate that such sequences must be systematically leveraged, and that NLP self-supervised pre-training is the appropriate representation learning technique.

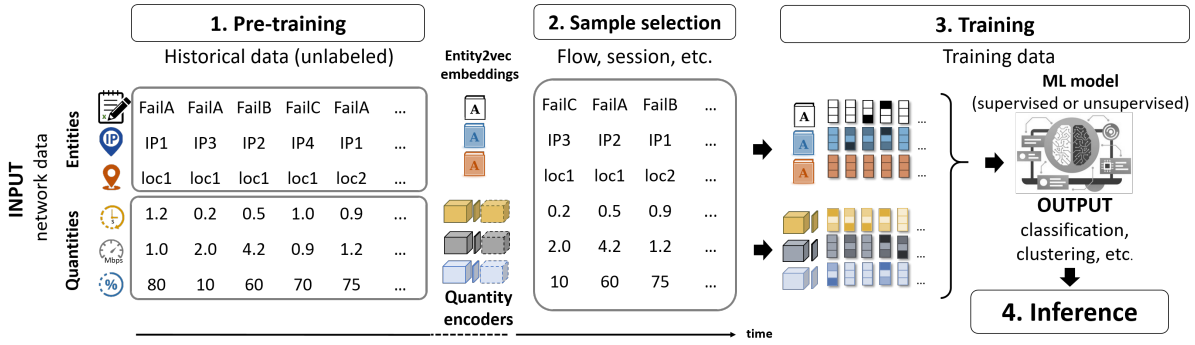

Figure 1: Generic bimodal pipeline.

We also acknowledge that not all network data is sequential or measured over time i.e., network topologies. However such data often pertains to entities e.g., node names or identifiers, for which sequence data is abundant e.g., routing advertisement, 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 is the learning from sequential data in NLP - words are nothing more than sequences of entities that follow each other. To perform a ML task, words must be first transformed into a numerical representation. This can be naïvely done using integer or one-hot encoding. Modern NLP models instead, either build or take as input word vector representations obtained through *self-supervised pre-trained models* created from large text corpora. A well known word *embedding* technique, which we use in this paper for its simplicity, is word2vec [25]. It transforms each word into a high dimensional vector, hence "embedding" it into an hyperspace. In practice, word embeddings are sometimes complemented with sub-word or character level embeddings [5, 19] i.e., with separate vector representations, for handling out-of-vocabulary words, misspellings, etc.

Building vector representations with word2vec. With word2vec, in a nutshell, a simple neural network with one hidden layer (whose dimensions are the embedding vectors size) is trained to predict a target word from its surrounding **context**. Word2vec thus does not need expensive or human-made labels, but rather cheaply and automatically 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, 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.

Emerging properties. Although trained to predict the next word in a sequence, vector representations learned by word embeddings exhibit interesting properties. A well known 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 with few adjustments one can observe that words with similar meaning fall in the same "positions" in each language vector [24]. This observation opened the way later to self-supervised language translation using only single-language corpora [23].

Conditions to apply language model pre-training. In NLP, word2vec and language model pre-training do not impose particular conditions the language must satisfy. However, moving away from natural language to any arbitrary sequence of named entities, we believe that *at least* two conditions must be satisfied. The foremost is the *(i) naming consistency*. Like words in natural language, network entities are expected to keep the same meaning[1]. Moreover, we require corpora *(ii) stability*: while this is true in natural language as adding a new word to the vocabulary is unfrequent, observing a new entity in network data is rather frequent. Drawing the proper conclusion from the above conditions, we 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.

[1]Exceptions may exist, e.g., the word "set". As such, contextual word embeddings like ELMo [26] have been devised to solve this problem.

2.3 A bimodal pipeline

In this section we sketch a generic bimodal pipeline, consisting of four steps namely *Pre-training*, *Sample selection*, *Training* and *Inference*, as shown in Figure 1.

Pre-training. Similarly to ML pre-training, the first phase of the proposed pipeline consists in leveraging huge amounts of unlabeled data to learn relevant representations from different data types. As shown in the leftmost part of Figure 1, the pipeline takes sequences of unlabeled network data as input. Quantities and entities are then fed to the most suitable representation learning pipelines, e.g., auto-encoder for quantities and word embeddings for entities.

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 ML task(s) will take as input. Unlike classic ML tasks whose samples are often clear e.g., a matrix of pixel values for CV or a sequence of strings for NLP, network-related ML tasks may have a variety of input samples. For example, if the goal is to classify IP addresses (or flows) as either malicious or benign, then the input sample should be a feature vector representation of the IP address (or 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 defined, its fixed-size vector representation is created by combining (*i*) the corresponding quantities (or their representations) and (*ii*) the learned entities representations. For the sake of simplicity, in the reminder of this paper entities' embbedings and quantities are combined by simple concatenation.

Training. When pre-training and sample selection are done, training a downstream ML task is rather straightforward. In an unsupervised use-case, one can simply cluster the vector representations. Likewise, in a supervised classification use-case, one can associate labels to the representations to train the classifier. Notice that, the more robust representations learned, the fewer labeled samples required for the downstream task [4, 28]. In the clickstream toy case presented herafter, we use a relatively small synthetic labeled dataset leveraging page visits of top 1000 Alexa ranking websites.

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

3 USE CASES

This section illustrates the advantages of embedding network entities, as opposed to using only quantities, using two illustrative use-cases. In showcasing our approach, we only focus on categorical data embeddings . Otherwise stated, we use word2vec pre-training for entities and leave raw unprocessed quantities as they are; while a better representation of quantities might exist, we leave it for future work.

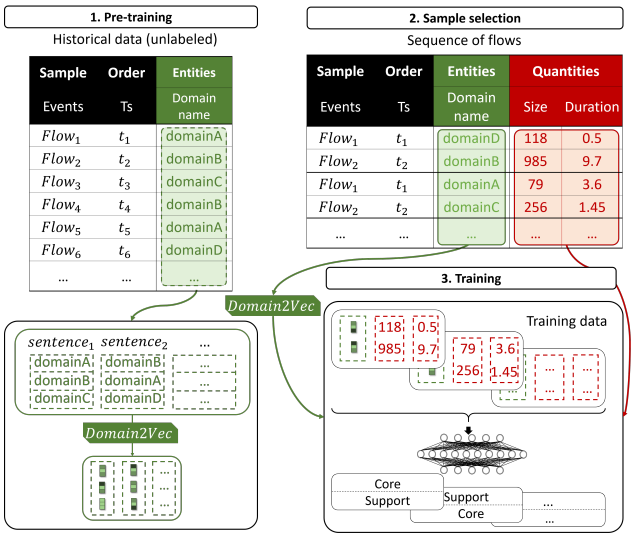

Figure 2: Bimodal pipeline for clickstream.

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, respectively 70 and 50 in the top 1000 Alexa dataset. Since the advent of encryption, an ISP collecting web traffic flow logs cannot infer (*i*) which flow belongs to which page, nor (*ii*) which flow queries the domain of the main page i.e., *core domain* and which query a necessary resource to render the page i.e., *support domains*. Such knowledge could be useful to estimate per-page Web quality of experience metrics from flow logs. Beyond the usefulness of the scenario itself, the two tasks above come 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 domain names (entities). Following our guidelines, the first step is pre-training. In this case, we train a *domain2vec* model later used to embed domain names. As mentioned earlier, 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 is a series of consecutive flows corresponding to the simultaneous query of an unknown number of web pages. A first ML task aims to "disentangle" flows by associating them to

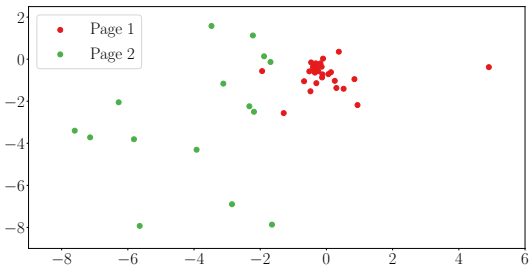

Figure 3: PCA visualization of domain embeddings

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 in the clickstream use case.

their Web page. A subsequent task is to classify flows as either *core* or *support*. We relied on a Gated Recurrent Unit (GRU) model for such binary task. In the following we qualitatively show how the entity-based representation helps solving these tasks by comparing different representations: quantity-only features (flow size and byte progression), word embedding only, and word & character embeddings.

For our evaluation, we consider a Web traffic dataset of 20k domains retrieved by downloading 10 times each of the Web pages from top 1,000 Alexa ranking, and recording 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 generates a vector representation of size 200. The supervised learning dataset instead consists of 60k similarly synthesized multisessions. We select around 900 pages for training and validation sets (50k multisessions). We test on 10k multisessions composed by the remaining *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 evaluation, Figure 3 plots a 2 component PCA representation of domain embeddings belonging to two web pages. Interestingly, without additional feature learning, 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 the *core*/*support* classification model, focusing on

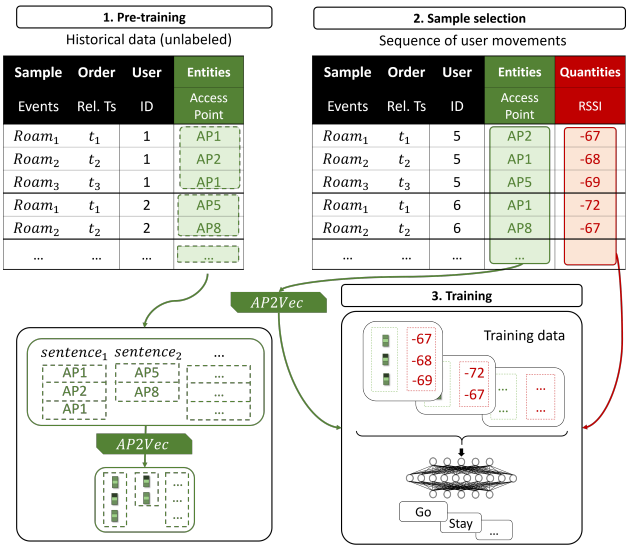

Figure 4: Bimodal pipeline for movement prediction.

precision and recall of the minor class, i.e., core domain. For this ML task, 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 case of multisession). For each multisession, we compute a precision and a recall in predicting the core domains and 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 outperformed the quantity-based one. Also, character embedding adds value compared to word embedding only. Note that in this case one-hot encoding is not a viable solution because the test set only contains unseen pages.

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, an important problem is to predict 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, 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. Following our bimodal pipeline, at pre-training, the RSSI and AP lists are grouped, this time, by user ID. The historical 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

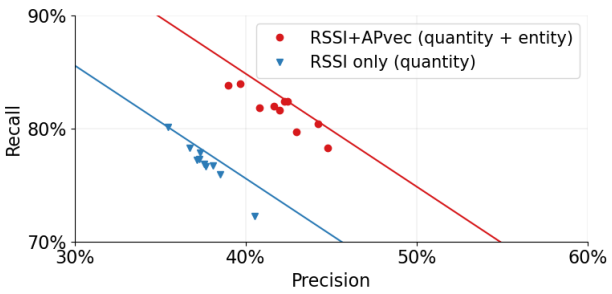

Figure 5: Comparison of movement prediction models.

with a 1D CNN using as input the concatenation of the RSSI series with the related AP embedding.

To evaluate the proposed bimodal pipeline, we consider a 5 days dataset of real network data with approximately 2k mobile terminals and 240k movement events across 80 different AP. The AP2vec pre-training is executed on 10 AP long sequences and generates an embedding vector of size 20. 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 is 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 helps 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 APs), 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 RSSI and AP2vec always lead to a slightly better model.

4 RELATED WORK

Recent work started learning alternative representations for entities. One of the first efforts is IP2Vec [27] which embeds source and destination IP addresses and ports with the objective of identifying IP addresses with similar behaviors. However, the embedding training is limited only to the 5-tuples, and hence does not exploit the the historical sequences of entities. With the goal of identifying malicious behaviors, DANTE [7] also leverages word2vec, but to embed ports that are sequentially tried by attackers. Similarly to DANTE, Darkvec [10] 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 example [11], close to the clickstream usecase, leverages word2vec to build user profiles from browsing historical data.

An alternative to learning representations is feature engineering, but the latter often ends up in using a single modality. Searching for the best representation, Traffic Refinery [3] seeks the right balance between a feature-selection that is effective i.e., accurate, and feasible i.e., deployable at line rate. nPrintML [15] 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.

Finally, with a few exceptions [1], we could not find examples in other domains beyond the classic (image,text) bimodality. It could be a matter of time before this issue is similarly tackled in other areas.

5 CONCLUSIONS

As a first step towards multimodality, we proposed a bimodal data representation of *entities* and *quantities*, in which historical sequences of *entities* are systematically transformed into vector representations using word embeddings. We showed the effectiveness of such representation through two toy examples as well as recent examples from the literature. As networks and systems in general are rife with sequential events, we believe the scope for potential applications of bimodal data representation learning are broader than the illustrative toy cases. Other relevant use cases in networking include sequences of IP addresses, BGP advertisements, routing events, alarms, etc. With the widespread of data-driven decision taking (e.g. finance, manufacturing), applications beyond networks are likewise numerous.

Yet, network data is more complex than co-occuring sequences of events and quantities illustrated in this paper. Hence, although useful as a conceptual framework, our bimodal representation 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 multimodality. Incorporating these pieces in the bigger puzzle of network data representation remains an interesting open research question for the networking community as a whole.

ACKNOWLEDGMENTS

We thank Shihao, Alexis Huet and Jonatan Krolikowski for their help on the two use cases. We are grateful to the anonymous reviewers for their feedback and quality review.

REFERENCES

[1] Bagus Tris Atmaja, Akira Sasou, and Masato Akagi. 2022. Survey on bimodal speech emotion recognition from acoustic and linguistic information fusion. *Speech Communication* (2022).

[2] Andrea Banino, Caswell Barry, Benigno Uria, Charles Blundell, Timothy Lillicrap, Piotr Mirowski, Alexander Pritzel, Martin J Chadwick, Thomas Degris, Joseph Modayil, et al. 2018. Vector-based navigation using grid-like representations in artificial agents. *Nature* 557, 7705 (2018), 429–433.

[3] Francesco Bronzino, Paul Schmitt, Sara Ayoubi, Hyojoon Kim, Renata Teixeira, and Nick Feamster. 2021. Traffic Refinery: Cost-Aware Data Representation for Machine Learning on Network Traffic. *Proc. ACM Meas. Anal. Comput. Syst.* 5, 3, Article 40 (dec 2021), 24 pages. https://doi.org/10.1145/3491052

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[5] Xinxiong Chen, Lei Xu, Zhiyuan Liu, Maosong Sun, and Huanbo Luan. 2015. Joint learning of character and word embeddings. In *Twenty-fourth international joint conference on artificial intelligence.*

[6] Radoslaw Martin Cichy, Aditya Khosla, Dimitrios Pantazis, Antonio Torralba, and Aude Oliva. 2016. Comparison of deep neural networks to spatio-temporal cortical dynamics of human visual object recognition reveals hierarchical correspondence. *Scientific reports* 6, 1 (2016), 1–13.

[7] Dvir Cohen, Yisroel Mirsky, Manuel Kamp, Tobias Martin, Yuval Elovici, Rami Puzis, and Asaf Shabtai. 2020. 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.* Springer-Verlag, 88–109.

[8] Christopher J Cueva and Xue-Xin Wei. 2018. Emergence of grid-like representations by training recurrent neural networks to perform spatial localization. *arXiv preprint arXiv:1803.07770* (2018).

[9] Mathias Franzius, Henning Sprekeler, and Laurenz Wiskott. 2007. Slowness and sparseness lead to place, head-direction, and spatial-view cells. *PLoS computational biology* 3, 8 (2007), e166.

[10] Luca Gioacchini, Luca Vassio, Marco Mellia, Idilio Drago, Zied Ben Houidi, and Dario Rossi. 2021. DarkVec: automatic analysis of darknet traffic with word embeddings. In *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies.* ACM.

[11] Roberto Gonzalez, Claudio Soriente, Juan Miguel Carrascosa, Alberto Garcia-Duran, Costas Iordanou, and Mathias Niepert. 2021. User Profiling by Network Observers. In *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies.* ACM.

[12] Torkel Hafting, Marianne Fyhn, Sturla Molden, May-Britt Moser, and Edvard I Moser. 2005. Microstructure of a spatial map in the entorhinal cortex. *Nature* 436, 7052 (2005), 801–806.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition.*

[14] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[15] Jordan Holland, Paul Schmitt, Nick Feamster, and Prateek Mittal. 2021. New Directions in Automated Traffic Analysis. *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security* (Nov 2021). https://doi.org/10.1145/3460120.3484758

[16] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* (2017).

[17] David H Hubel and Torsten N Wiesel. 1959. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology* 148, 3 (1959), 574.

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

[19] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *Thirtieth AAAI conference on artificial intelligence.*

[20] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR.*

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2017. Imagenet classification with deep convolutional neural networks. *Commun. ACM* 60, 6 (2017), 84–90.

[22] Norbert Kruger, Peter Janssen, Sinan Kalkan, Markus Lappe, Ales Leonardis, Justus Piater, Antonio J Rodriguez-Sanchez, and Laurenz Wiskott. 2012. Deep hierarchies in the primate visual cortex: What can we learn for computer vision? *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2012), 1847–1871.

[23] Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043* (2017).

[24] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168* (2013).

[25] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems* 26 (2013).

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

[27] Markus Ring, Alexander Dallmann, Dieter Landes, and Andreas Hotho. 2017. IP2Vec: Learning Similarities Between IP Addresses. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW).* 657–666. https://doi.org/10.1109/ICDMW.2017.93

[28] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. 2020. Rethinking few-shot image classification: a good embedding is all you need?. In *European Conference on Computer Vision.* Springer, 266–282.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).