ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

INFORMATION TECHNOLOGY

UDC 004.8 **DOI: 10.20998/2079-0023.2019.01.09**

R. V. SHAPTALA, G. D. KYSELEV

USING GRAPH EMBEDDINGS FOR WIKIPEDIA LINK PREDICTION

Link prediction is an important area of study in network analysis and graph theory which tries to answer the question of whether two nodes in the graph might have an association in the future. Nowadays, graphs are ubiquitously present in our lives (social networks, circuits, roads etc.), which is why the problem is crucial to the development of intelligent applications. In the past, there have been proposed methods of solving link prediction problem through algebraic formulations and heuristics, however, their expressive power and transferability fell short. Recently, graph embedding methods have risen to popularity because of their effectiveness and the ability to transfer knowledge between tasks. Inspired by the famous in machine learning and natural language processing research Word2Vec approach, these methods try to learn a distributed vector representation, called an embedding, of graph nodes. After that a binary classifier given a pair of embeddings predicts the probability of the existence of a link between the encoded nodes. In this paper, we review several graph embedding approaches for the problem of Wikipedia link prediction, namely Wikipedia2vec, Role2vec, AttentionWalk and Walkets. Wikipedia link prediction tries to find pages that should be interlinked due to some semantic relation. We evaluate prediction accuracy on a hold-out set of links and show which one proves to be better at mining associations between Wikipedia concepts. The results include qualitative (principal component analysis dimensionality reduction and visualization) and quantitative (accuracy) differences between the proposed methods. As a part of the conclusion, further research questions are provided, including new embedding architectures and the creation of a graph embedding algorithms benchmark.

Keywords: graph embeddings, link prediction, Wikipedia2vec, Role2vec, AttentionWalk, Walklets, principle component analysis.

Р. В. ШАПТАЛА, Г. Д. КИСЕЛЬОВ

ВИКОРИСТАННЯ ВЕКТОРНИХ ПРЕДСТАВЛЕНЬ ГРАФІВ ДЛЯ ПРОГНОЗУВАННЯ ЗВ’ЯЗКІВ У WIKIPEDIA

Прогнозування зв'язків є важливою областю дослідження в аналізі мереж та теорії графів, яка намагається відповісти на питання, чи можуть два вузли у графі в майбутньому мати зв’язок. На сьогоднішній день графи повсюдно присутні у нашому житті (соціальні мережі, електротехніка, дороги і т.д.), тому проблема має вирішальне значення для розвитку інтелектуальних додатків. У минулому були запропоновані методи вирішення задачі прогнозування зв’язків за допомогою алгебраїчних формулювань і евристик, однак їхня виразність і переносимість не були задовільними. Останнім часом методи побудови векторних представлень зросли у популярності через їх ефективність і здатність передавати знання між завданнями. Натхненний знаменитим в машинному навчанні та обробці природних мов дослідницьким підходом Word2Vec, ці методи намагаються вивчити розподілене векторне представлення. Після цього бінарний класифікатор, заданий парою таких векторів, прогнозує ймовірність існування зв'язку між закодованими вузлами. У даній роботі ми розглянемо декілька підходів до вбудовування графіків для проблеми прогнозування зв’язків у Wikipedia, а саме Wikipedia2vec, Role2vec, AttentionWalk та Walkets. Прогнозування посилань у контексті Wikipedia – це знаходження сторінок, які пов'язані через певні смислові відносини. Ми оцінюємо точність прогнозування на відокремленому наборі зв’язків і показуємо, який з методів краще знаходить асоціації між сутностями у Вікіпедії. Отримані результати включають якісні (метод головних компонентів для зменшення розмірності та візуалізації) і кількісні (точність) відмінності між запропонованими методами. У рамках висновку наводяться подальші дослідницькі питання, включаючи нові архітектури побудови векторних представлень та створення загальноприйнятого тесту ефективності таких представлень.

Ключові слова: векторні представлення даних, прогноз зв’язків, Wikipedia2vec, Role2vec, AttentionWalk, Walklets, метод головних компонентів.

Р. В. ШАПТАЛА, Г. Д. КИСЕЛЕВ

ИСПОЛЬЗОВАНИЕ ВЕКТОРНЫХ ПРЕДСТАВЛЕНИЙ ГРАФОВ ДЛЯ ПРОГНОЗИРОВАНИЯ СВЯЗЕЙ В WIKIPEDIA

Прогнозирование связей является важной областью исследования в анализе сетей и теории графов, которая пытается ответить на вопрос, могут два узла в графе в будущем иметь связь. На сегодняшний день графы повсеместно присутствуют в нашей жизни (социальные сети, электротехника, дороги и т.д.), поэтому проблема имеет решающее значение для развития интеллектуальных приложений. В прошлом были предложены методы решения задачи прогнозирования связей с помощью алгебраических формулировок и эвристик, однако их выраженность и переносимость ни были удовлетворительными. В последнее время методы построения векторных представлений выросли в популярности из-за их эффективности и способности передавать знания между задачами. Вдохновленный знаменитым в машинном обучении и обработке естественных языков исследовательским подходом Word2Vec, эти методы пытаются изучить распределено векторное представление. После этого бинарный классификатор, заданный парой таких векторов, прогнозирует вероятность существования связи между закодированными узлами. В данной работе мы рассмотрим несколько подходов к построению векторных представлений для проблемы прогнозирования связей в Wikipedia, а именно Wikipedia2vec, Role2vec, AttentionWalk и Walkets. Прогнозирование связей в контексте Wikipedia – это нахождение страниц, которые связаны через определенные смысловые отношения. Мы оцениваем точность прогнозирования на отдельном наборе связей и показываем, какой из методов лучше находит ассоциации между сущностями в Википедии. Полученные результаты включают качественные (метод главных компонент для уменьшения размерности и визуализации) и количественные (точность) различия между предлагаемыми методами. В рамках заключении приводятся дальнейшие исследовательские вопросы, включая новые архитектуры построения векторных представлений и создание общепринятого теста эффективности таких представлений.

Ключевые слова: векторные представления графов, прогноз связей, Wikipedia2vec, Role2vec, AttentionWalk, Walklets, метод главных компонент.

Introduction. Networks and graphs have become ubiquitously important to model difficult systems that consist of various elements. Graph data science has a large number of applications in various fields like logistics, social networks, recommendation engines, and communication networks. There have been a lot of research in the area of the possibility to predict new links between elements in the topology of the graph based on the properties of its elements. Such a task is called link prediction and is defined as the problem of predicting new relationships in networks. Link prediction's goal is to find the initial rules of the graph link formation by inferring lost or possible relationships, given currently observed connections. The area is growing fast and is becoming more and more interesting as a research vector since it can help us predict how real-life networks will progress and evolve in time [1].

One of the applications of graphs and an example of complex networks are web-scale knowledge bases [2]. They provide a representation of world knowledge that is structured, with projects such as the Google Knowledge Vault [3], Freebase [4] and DBPedia [5]. These technologies are at the core of a wide range of applications such as question answering, recommender systems and chatbots. Unfortunately, these knowledge bases are incomplete because of the complexity of our world. That is why predicting missing entries or link prediction is one of the main problems in knowledge engineering. Knowledge bases encode data as a directed graph with edges (links, relations) between nodes (concepts, entities). The topological structure and nature among the relations present in these bases often make the taks of filling in the missing links of a knowledge base possible. The idea behind link prediction is the automatic search for such regularities.

There are two types of approaches that are usually used to define models for graph-based problems [6]. The first one works with the initial graph adjacency matrix, while the second – with an inferred vector space. The popularity of the last approach has gradually increased lately. They try to represent the graph in a vector space that is going to preserve its properties. Having such an encoding is extremely convenient in the graph-related problems. The vectors are used as inputs (features) to a machine learning algorithm which parameters are trained based on the dataset. This helps negate the need for difficult classification algorithms which work directly with the graph.

However, the dimensions of the trained vectors become an additional hyperparameter and searching for an optimal one can be difficult. For example, higher dimensionality might increase the reconstruction precision but will have higher time and space complexities. The choice can also be domain-specific depending on the task: for example, lower number of dimensions might result in better link prediction accuracy if the model only captures local relations between entities [6].

Preliminaries. A graph G(V, E) is a collection of $V = \{v_1, \dots, v_n\}$ nodes and $E = \{e_{ij}\}_{i,j=1}^{n}$ edges. The adjacency matrix S of graph G contains indicators associated with each edge in the following way: $s_{ij} = 1$ if v_i and v_j are connected to each other, and s_{ij} = 0 otherwise. For undirected graphs, $s_{ij} = s_{ji}\ \forall i,j \in \{1, \dots, n\}$.

Given a graph G(V, E), a graph embedding is a mapping $f\colon v_i \to w_i \in \mathbb{R}^d\ \forall i \in \{1, \dots, n\}$ such that $d \ll |V|$ and the function f retains some similarity notion defined on graph G.

Consequently, a graph embedding encodes each node in a low-dimensional feature vector that can retain the relations between nodes.

Graph embeddings. In this section we describe evaluated graph embedding approaches.

Random walks are at the core of numerous existing graph embedding methods. Since such approaches have numerous problems that arise from their exploitation of random walks (like the features that can not transfer knowledge to other nodes and networks as they are unique to each entity. Role2Vec framework tries to overcome this drawback by the use of attributed random walks. This algorithm was chosen because it is a basis for generalizing other similar methods like DeepWalk, node2vec, and many others that are based on random walks. The proposed framework helps these methods be more applicable for both transductive and inductive learning as well as for use on graphs with other features (if they exist) [7]. This is accomplished by learning functions that are applicable to unseen entities and networks. The authors show that Role2vec is more efficient in terms of predictive performance as well as requires less space than other methods on a variety of graphs. Role2Vec uses the extensible notion of attributed random walks that is not connected to a specific node but is instead based on a function that maps a node feature vector to a class, so that two nodes belong to the same class if they are topologically similar. Role2vec provides several valuable advantages to any method that is built upon it. Firstly, it is naturally inductive as the learned embeddings generalize to new entities and across networks and therefore might be used for transfer learning. Secondly, authors claim that their approach is able to capture structural similarity more efficiently. Thirdly, the Role2vec framework is way more space-efficient since representations are learned for classes (not nodes) and consequently require less space than existing methods. Fourthly, the proposed framework has an ability to work with graphs with features (if such exist or are available).

Graph embedding methods encode nodes in a continuous vector space, capturing various classes of information present in the network. These methods have numerous hyperparameters (for example the length of a random walk) which have to be manually tuned for every graph. AttentionWalk is a method of graph embedding where previously fixed hyperparameters are replaced with trainable ones that are automatically learned via backpropagation [8]. The authors propose an attention model on the power series of the transition matrix, which decides where to take the next walk in order to optimize a long-term goal. Different to other attention models, the AttentionWalk uses attention parameters only on the training data itself (the random walk), while during model inference there are no attention layers. The authors did a series of tests on link prediction tasks, trying to produce embeddings that capture the graph structure, transferring the representation to unseen information. It is also claimed that AttentionWalk improves state-of-the-art results on a set of real-world graph datasets, for example collaboration, biological, and social networks. The final result of this approach is that automatically-learned attention parameters tend to correspond with the optimal choice of hyper-parameters that are manually tuned in other methods.

Another approach which is of particular interest to us is called Walklets [9], a novel method for learning multiscale representations of nodes in a graph. These vectors explicitly encode multiscale relationships in a way that is analytically derivable compared to previous works. The proposed method creates these multiscale relationships by subsampling random walks of different length on the nodes of a graph. By skipping over steps in each random walk, Walklets generates a different training dataset than similar approaches. More specifically, it creates a corpus of node pairs which are reachable via paths of a fixed length. This corpus is then used to find a set of hidden representations, each of which encodes successively higher order relationships from the adjacency matrix. The authors demonstrate the efficiency of Walklets' hidden representations on several multi-label graph classification tasks for social applications. Their results claim that Walklets outperforms other methods based on neural matrix factorization. One of the most important benefits of Walklets is that it is an online learning algorithm, so it can scale to networs with an enormous number of nodes and links.

A different, yet powerful algorithm that can be used for Wikipedia link prediction is Wikipedia2vec – an open source tool for learning embeddings of words and entities from Wikipedia [10]. Not only does this tool enable researchers to easily obtain high-dimensional embeddings of words and entities from a Wikipedia dump, it also provides the source code, documentation, and pretrained vectors for twelve most popular languages at http://wikipedia2vec.github.io. The learned embeddings can easily be applied via transfer learning for natural language processing (NLP) models. The tool can be installed via Python programming language package repository PyPI. The pretrained embeddings have been learned by iterating over entire Wikipedia pages and joint optimization of three different submodels: model of Wikipedia graph, which learns entity embeddings by predicting neighboring entities in Wikipedia's page network – an undirected graph whose nodes are entities and edges represent links between entities, based on each entity in Wikipedia (it does not matter if both pages link to each other or only one of them references another one – the link is created anyway); word-based skip-gram model, which learns word embeddings by predicting neighboring words for each word in a text contained on a Wikipedia page; anchor context model, which aims to place similar words and entities near one another in the vector space, and to create interactions between embeddings of words and those of entities. Here, we obtain referent entities and their neighboring words from links contained in a Wikipedia page, and the model learns embeddings by predicting neighboring words given each entity.

These three submodels are all inspired by the skip-gram model [11], which is a neural network model with a training objective to find embeddings that are useful for predicting context items (i.e., neighboring words or entities) given a target item.

To predict links between two nodes in a graph we use a simple one hiden layer perceptron on the concatenation of the embeddings of both nodes. The final classification task is trained using ADAM [12] optimization algorithm with the learning rate of 0.01 and 100 hidden layer units.

Evaluation and results.

We evaluated the described approaches on the SNAP Wikispeedia [13] navigation paths dataset. This dataset has a set of Wikipedia links, collected through the human-computation game, called Wikispeedia. In there, users are asked to navigate from a starting Wikipedia node to a given article, through clicking Wikipedia links. A condensed version of Wikipedia (4,604 articles) is used.

For our project, 107444 links were used as positive examples and the same quantity was generated as negative examples. Thus, the dataset is balanced and we can use accuracy to measure the performance of the implemented approaches. 20 % of the data was held out for testing and the results are presented on this test set.

Quantitative results of our evaluation are summarized in Table 1. Walklets significantly outperforms every other approach that we tested due to the subsampling that is inherent in the algorithm, capturing not only first-order information, but also encoding the relations between nodes further from the start of the random walk.

Table 1 – Evaluated embeddings link prediction accuracy

Embeddings	Accuracy
Role2vec	0.723
AttentionWalk	0.699
Walklets	0.877
Wikipedia2vec	0.734

To provide some qualitative results, we have also tried plotting the resulting embeddings. Since all of the tested approaches provide high-dimensional representations, the first problem that arises is to reduce these dimensions to human-readable form. For that we use principal component analysis (PCA) with the number of principle components set to 2. In our case, PCA transforms the data to a new coordinate system where the highest variance by some projection of the data comes to lie on the first coordinate and the second highest variance on the second coordinate. From Fig. 1–4 it can be seen that AttentionWalk could not capture meaningful information, since there are no well-defined clusters on the visualization. Role2vec and Wikipedia2vec managed to group similar concepts in several clusters, however Walklets show a better space division than them. This correlates with the quantitative results that were shown previously.

Conclusions. In this paper, we reviewed several graph embedding approaches for the problem of Wikipedia link prediction, namely Wikipedia2vec, role2vec, AttentionWalk and Walkets. Qualitative and quantitative results show that Walklets due to its implicit multiscale relationship capture system have more expressive power for the given problem.

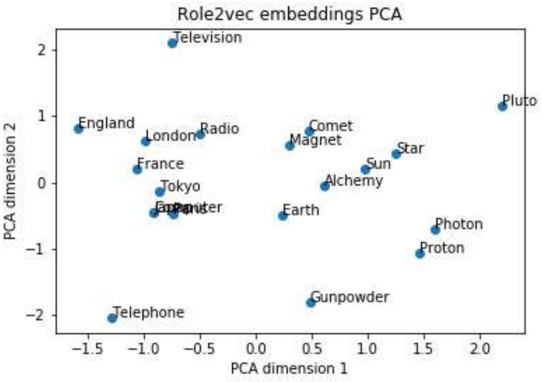

Fig. 1. Role2vec embeddings reduced to 2-D by PCA

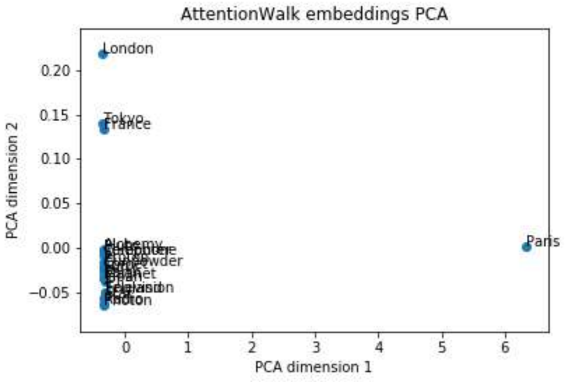

Fig. 2. AttentionWalk embeddings reduced to 2-D by PCA

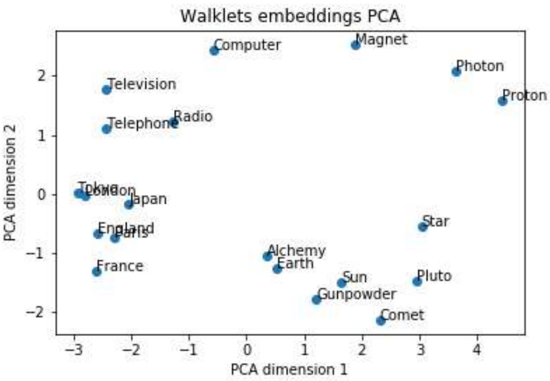

Fig. 3. Walklets embeddings reduced to 2-D by PCA

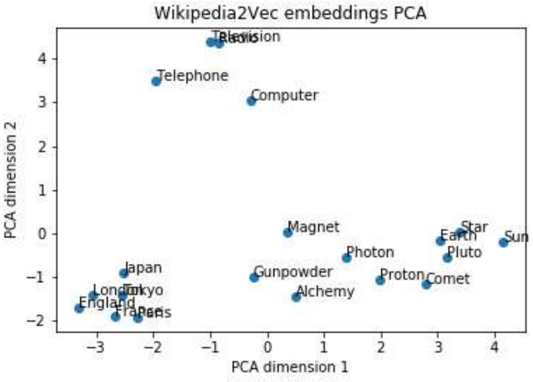

Fig. 4. Wikipedia2Vec embeddings reduced to 2-D by PCA

We consider the following research directions valid for future work: the creation of a standard benchmark dataset for link prediction of sufficient size to test accuracy, speed and scalability of graph embedding approaches; experimenting with new architectures, that would capture more information inherent to the link prediction problem, since our work did not achieve perfect prediction accuracy.

References

1. Martínez V., Berzal F., Cubero J. C. A survey of link prediction in complex networks. *ACM Computing Surveys (CSUR)*. 2017. Vol. 49, № 4. Article 69.
2. Minervini P., Costabello L., Muñoz E., Nováček V., Vandenbussche P. Y. Regularizing knowledge graph embeddings via equivalence and inversion axioms. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. *Vol. 10534*. Cham: Springer, 2017. P. 668–683.
3. Dong X., Gabrilovich E., Heitz G., Horn W., Lao N., Murphy K., Strohmann T., Sun S., Zhang W. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. New York: ACM, 2014. P. 601–610.
4. Bollacker K., Evans C., Paritosh P., Sturge T., Taylor J. Freebase: a collaboratively created graph database for structuring human knowledge. *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. Vancouver: ACM, 2008. P. 1247–1250.
5. Auer S., Bizer C., Kobilarov G., Lehmann J., Cyganiak R., Ives Z. Dbpedia: A nucleus for a web of open data. *The semantic web*. Berlin, Heidelberg: Springer, 2007. P. 722–735.
6. Goyal P., Ferrara E. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*. 2018. Vol. 151. P. 78–94.
7. Ahmed N. K., Rossi R., Lee J. B., Willke T. L., Zhou R., Kong X., Eldardiry H. Learning role-based graph embeddings. *StarAI workshop*, *IJCAI 2018*. *arXiv preprint arXiv:1802.02896*. 2018.
8. Abu-El-Haija S., Perozzi B., Al-Rfou R., Alemi A. A. Watch your step: Learning node embeddings via graph attention. *Advances in Neural Information Processing Systems*. *Vol. 31*. Curran Associates, Inc. Publ., 2018. P. 9180–9190.
9. Perozzi B., Kulkarni V., Chen H., Skiena S. Don't Walk, Skip! Online Learning of Multi-scale Network Embeddings. *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. New York: ACM, 2017. P. 258–265.
10. Yamada I., Asai A., Shindo H., Takeda H., Takefuji Y. Wikipedia2Vec: An Optimized Implementation for Learning Embeddings from Wikipedia. *arXiv preprint arXiv:1812.06280*. 2018.
11. Mikolov T., Sutskever I., Chen K., Corrado G. S., Dean J. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*. *Vol. 2*. Curran Associates, Inc. Publ., 2013. P. 3111–3119.
12. Kingma D. P., Ba J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. 2014.

13. West R., Pineau J., Precup D. June. Wikispeedia: An online game for inferring semantic distances between concepts. *Twenty-First International Joint Conference on Artificial Intelligence*. Pasadena, 2009. P. 1598–1603.

References (transliterated)

1. Martínez V., Berzal F., Cubero J. C. A survey of link prediction in complex networks. *ACM Computing Surveys (CSUR).* 2017, vol. 49, no. 4, article 69.
2. Minervini P., Costabello L., Muñoz E., Nováček V., Vandenbussche P. Y. Regularizing knowledge graph embeddings via equivalence and inversion axioms. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases.* Cham, Springer Publ., 2017, vol. 10534, pp. 668–683.
3. Dong X., Gabrilovich E., Heitz G., Horn W., Lao N., Murphy K., Strohmann T., Sun S., Zhang W. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining.* New York, ACM Publ., 2014, pp. 601–610.
4. Bollacker K., Evans C., Paritosh P., Sturge T., Taylor J. Freebase: a collaboratively created graph database for structuring human knowledge. *Proceedings of the 2008 ACM SIGMOD international conference on Management of data.* Vancouver, ACM Publ., 2008, pp. 1247–1250.
5. Auer S., Bizer C., Kobilarov G., Lehmann J., Cyganiak R., Ives Z. Dbpedia: A nucleus for a web of open data. *The semantic web.* Berlin, Heidelberg, Springer Publ., 2007, pp. 722–735.
6. Goyal P., Ferrara E. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*. 2018, vol. 151, pp.78–94.
7. Ahmed N. K., Rossi R., Lee J. B., Willke T. L., Zhou R., Kong X., Eldardiry H. Learning role-based graph embeddings. *StarAI workshop, IJCAI 2018*. *arXiv preprint arXiv:1802.02896*. 2018.
8. Abu-El-Haija S., Perozzi B., Al-Rfou R., Alemi A. A. Watch your step: Learning node embeddings via graph attention. *Advances in Neural Information Processing Systems. Vol. 31.* Curran Associates, Inc. Publ., 2018, pp. 9180–9190.
9. Perozzi B., Kulkarni V., Chen H., Skiena S. Don't Walk, Skip! Online Learning of Multi-scale Network Embeddings. *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining.* New York, ACM Publ., 2017, pp. 258–265.
10. Yamada I., Asai A., Shindo H., Takeda H., Takefuji Y. Wikipedia2Vec: An Optimized Implementation for Learning Embeddings from Wikipedia. *arXiv preprint arXiv:1812.06280*. 2018.
11. Mikolov T., Sutskever I., Chen K., Corrado G. S., Dean J. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems. Vol. 2.* Curran Associates, Inc. Publ., 2013, pp. 3111–3119.
12. Kingma D. P., Ba J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. 2014.
13. West R., Pineau J., Precup D. June. Wikispeedia: An online game for inferring semantic distances between concepts. *Twenty-First International Joint Conference on Artificial Intelligence*. Pasadena, 2009, pp. 1598–1603.

Received 13.05.2019

Відомості про авторів / Сведения об авторах / About the Authors

Шаптала Роман Віталійович (***Шаптала Роман Витальевич***, ***Shaptala Roman Vitaliyovych***) – аспірант кафедри системного проектування Навчально-наукового комплексу «Інститут прикладного системного аналізу», Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»; м. Київ, Україна; ORCID: https://orcid.org/0000-0002-4367-5775; e-mail: r.shaptala@gmail.com

Кисельов Геннадій Дмитрович (***Киселев Геннадий Дмитриевич***, ***Kyselev Gennadiy Dmytrovych***) – кандидат технічних наук, старший науковий співробітник, доцент, заступник завідувача кафедри системного проектування Навчально-наукового комплексу «Інститут прикладного системного аналізу», Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»; м. Київ, Україна; ORCID: https://orcid.org/0000-0003-2682-3593; e-mail: kiselev@cad.kiev.ua