

Establishing Pattern Sequences Using Artificial Neural Networks with an Application to Organizational Patterns

Viktor Matovič¹, Valentino Vranić¹

Institute of Informatics, Information Systems and Software Engineering, Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava, Ilkovičova 2, 84216 Bratislava 4, Slovakia

Abstract

Explicit relationships included in textual description of patterns are insufficient to establish pattern sequences. This article brings a new method of establishing pattern sequences using artificial neural networks. The method is based on extracting implicit relationships between organizational patterns using the softmax regression model by Zhang et al., which are learned by the artificial neural network and encoded in its weights. The implementation of the method is based on the mini-batch stochastic gradient descent method by Zhang et al. and the Keras application programming interface of the TensorFlow framework. To evaluate the method, we conducted four experiments on selected organizational patterns documented by Coplien and Harrison. Two pattern sequences were established in each experiment. For each pattern sequence, a pattern story was created in order to check its meaningfulness.

Keywords

organizational pattern, pattern sequence, pattern relationships, artificial neural networks, softmax regression

1. Introduction

The idea of patterns came to software development from the work of Alexander in building architecture [1, 2]. There, they are not used only to design software, but also to build and maintain organizations of people who develop software. Such patterns are known as organizational patterns of software development [3]. Patterns have been identified in many other areas, such as teaching [4], drama [5, 6, 7, 8], or games [9].

Closely related patterns constitute pattern languages. While pattern languages must be understood before the expected order of application of patterns in pattern sequences can be determined [10], explicit relationships included in textual description of patterns are insufficient to establish pattern sequences because the types of these relationships do not determine the expected order of patterns [11]. Implicit relationships between patterns can be used to determine the expected order of patterns in pattern sequences. These can be extracted from textual


SQAMIA 2023: Workshop on Software Quality Analysis, Monitoring, Improvement, and Applications, September 10–13, 2023, Bratislava, Slovakia

✉ xmatovicv@stuba.sk (V. Matovič); vranic@stuba.sk (V. Vranić)

🌐 <http://fiit.sk/~vranic/> (V. Vranić)

🆔 0000-0002-8083-232X (V. Matovič); 0000-0001-9044-4593 (V. Vranić)

© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

descriptions of patterns [12], and this can be done in an automated way using machine learning methods and techniques [13].

This article brings a new method of establishing pattern sequences using artificial neural networks. The method is based on computing the probability of occurrence of patterns in pattern sequences. Consequently, the method results in *expected* (probable) pattern sequences.

The rest of this article is structured as follows. Section 2 presents existing approaches to extracting implicit relationships between patterns, which can be used to establish pattern sequences. Section 3 explains the core of our method of establishing pattern sequences. Section 4 explains the aspects of the artificial neural network used in our method. Section 5 explains the implementation of the method. Section 6 presents the experiments. Section 7 discusses the results. Section 8 relates this article to the work done by others. Section 9 concludes the article.

2. Extracting Implicit Relationships Between Patterns

Waseeb et al. [13] showed that the strength of the relationship between patterns can be calculated from the co-occurrence of vocabulary among the patterns. Waseeb et al. [13] used textual descriptions of thirty organizational patterns documented by Coplien and Harrison [3] as a source for mining hidden relationships.

Kaliyar et al. [14] presents a language model based on the BERT language model, which can be used to understand links to other patterns using the context in which these links occur, thus shifting from lexical to syntax analysis.

Yu et al. [15] present another model based on the BERT language model, which can be used to understand the semantic meaning of the text in a pattern story before or after applying the pattern. After applying a particular pattern, the resulting context could be used to identify the next applicable pattern.

The softmax regression model by Zhang et al. [16] can be used to compute the probability of the event. If applied to organizational pattern descriptions, the sorted probabilities from the output of the softmax regression model could be used to establish pattern sequences. According to Zhang et al. [16], their softmax regression model can be implemented as an artificial neural network.

One of the advantages of using artificial neural networks to calculate the probability that a given sequence of terms describes an organizational pattern is that they can work well with learned representations of organizational patterns. Learned representations are widely considered more effective for efficiently applying deep learning methods.

An advantage of using this approach lies in its ability to produce meaningful pattern sequences based on probabilities of use of patterns in them and the ability to establish sequences of patterns from various pattern languages, possibly from different domains. Another advantage of using this approach is massive use of patterns cannot be observed.

A disadvantage of using an artificial neural network to establish pattern sequences is that they tend to overfit a training data set if less data is provided for training. To overcome model overfit on the training dataset, a Bayesian neural network can be used as the specific architecture of the artificial neural network. As discussed by Jospin [17] or Zhang et al. [16], at least 10 000–15 000 rows in training tables must be used to train a model to get a good generalization.

3. The Core of the Method

The softmax regression model by Zhang et al. is designed to produce soft predictions. In our method, soft predictions mean probability fragments of textual descriptions of organizational patterns can be ascribed to the names of the organizational patterns. According to Zhang et al. [16], their softmax regression model can be implemented as a single-layer neural network. Still, to strengthen the model, this does not inherently mean the neural network implementation must consist of a single hidden layer. Because of this, nine hidden processing layers were used in the neural network to work with bigrams. The neural network used to work with trigrams consisted of four hidden processing layers.

Artificial neural networks can be used to compute probabilities of the use of patterns in pattern sequences. The network used in this article was designed to classify term frequencies of terms in textual descriptions of patterns to names of these patterns. Probabilities of classifying term frequencies of terms to names of organizational patterns can be used to establish patterns into sequences. Textual descriptions of patterns do not have to refer to each other because the network tries to learn relationships between patterns by learning the content of their textual descriptions.

The softmax regression model by Zhang et al. produces vectors of probabilities for each pattern in the dataset. The vector of probabilities for each row of the dataset displayed in Table 1, i.e., $p_i = (prob_1, \dots, prob_n)$, predicts the probability that fragments of a textual description can be ascribed to an organizational pattern. The index of the highest probability $\text{argmax}_i prob_i$ points to the organizational pattern, which can be described by the first row of the frequency table. The index of the probability in this vector points to the event that the sentence from the first row of the table describes the problem solved by the pattern at the row number equal to this index. Different vectors with probabilities are produced in each run of the artificial neural network.

The output from the softmax regression model by Zhang et al. is computed by the softmax operation, idea for which was originally introduced by Bridle [18]. Outputs from output layer are inputs to the softmax activation function implementing Bridle's softmax operation in an artificial neural network. The softmax activation function was invented by Luce, who found that the softmax operation ensures that the output from this function sums up to 1 and will always be between 0 and 1 [16]. This way, the output from the softmax regression model conforms to the axiomatic definition of probability.

Here, a new method of establishing pattern sequences from their descriptions based on the softmax regression model by Zhang et al. is proposed. The input to the method is a set of pattern descriptions. The output is a set of the expected pattern sequences. The method consists of the following steps:

1. Split the pattern descriptions into vectors with two or more words. The number of vector components depends on your estimate.
2. Calculate term frequencies of the terms in the vectors gained in step 1.
3. Create a table so its first column contains the pattern's name (see Table 1 for an example).
4. Use the softmax regression model by Zhang et al. to compute the probability that a sentence constructed by concatenating pairs and triplets of the terms in the first row of

the table describes the problem that the pattern whose name is in the first column of this table tries to solve.

5. Sort the probability values gained in step 4 in descending order.
6. Identify the highest probability in all vectors. It points to the first vector in the pattern sequence. A pattern expected to be used as the first pattern in the pattern sequence is at the row equal to the index of this probability.
7. The second highest probability points to the second pattern in the sequence. Any subsequent probability in the ordered list points to the next pattern expected to be used in the sequence.
8. Verify that the pattern sequence established is meaningful. If this pattern sequence is meaningful, this pattern sequence is expected.

4. Artificial Neural Network Aspects

In the artificial neural network used in our method, each artificial neuron is expected to store information about the frequencies of words the organizational pattern was documented with. Each artificial neuron is also expected to store more information about the actual value of parameters, so that they could be later optimized with the gradient-descent optimization method and objective functions.

Artificial neural networks consist of input, hidden, and output layers. The output layer used in our method is a fully connected layer that produced probabilities of applying patterns in sequences.

In our method, artificial neural networks learn the structure of textual descriptions of organizational patterns encoded with learned parameters (numerically encoded in weights) and hyper-parameters. Hyper-parameters can also be represented in decisions to use various weight optimization algorithms or regularization strategies. Batch normalization from Ioffe [19] was used as a regularization strategy for experiments with bigrams and trigrams. Learned parameters are numerical weights and biases, which are always initialized randomly with weight initializers and subsequently optimized by the processing of the artificial neural network. Weight initializers of the Keras application programming interface of the TensorFlow platform were used in experiments. For both experiments, the size of the batches and the number of epochs were manipulated as hyper-parameters of networks used.

According to Zhang et al. [16], the parameterization cost of the output layer $O(o * x)$ can be minimized with an additional constrain factor: $O((o*x) / \text{hyper_parameter})$.

No guarantee exists that outputs produced by the output layer conform to the axiomatic definition of probability. The softmax activation function by Zhang et al. [16] was used to transform output from output layer into the probability of applying patterns in pattern sequence. Although the activation function is nonlinear, artificial neurons are defined as linear transformations, so the softmax regression model by Zhang et al. is implemented here as a linear regression model. Biases b_i in the definitions of the affine transformation from Table 1 are normally distributed, i.e., they follow the normal distribution with mean 0 and standard deviation equal to 1 $\mathcal{N}(\mu = 0, \sigma^2 = 1)$.

Table 1

The dataset for the artificial neural network.

		Feature	x_1	x_2	x_3	x_4
			Bigram TF-IDF	Bigram TF-IDF	Bigram TF-IDF	Bigram TF-IDF
Logit	Pattern	Vector	project+must	must+balance	people+critic	new+employee
0_1	Apprenticeship	(1,0,0,0,0)	0.65	1.01	1.01	1.01
0_2	Team Per Task	(0,1,0,0,0)	1.29	1.01	1.01	1.01
0_3	Size The Organization	(0,0,1,0,0)	1.29	1.01	1.01	1.01
0_4	Divide And Conquer	(0,0,0,1,0)	0.65	1.01	1.01	1.01
0_5	Architecture Team	(0,0,0,0,1)	0.65	1.01	1.01	1.01

Output from the artificial neural network for each organizational pattern is a vector of conditional probabilities ($prob_1, \dots, prob_n$). These probabilities are called conditional because they are defined as probabilities of the events that fragments of textual descriptions can be ascribed to given organizational patterns, formally: $P(y = \text{organizational pattern} \mid \text{observed term frequencies})$. According to Zhang et al. [16], the probabilities must be computed with minimum cross-entropy loss, thus minimizing surprises in model output. The purpose of using the softmax regression model by Zhang et al. was to find the best values of weights and biases such that they maximize the likelihood of observed frequencies of words in textual descriptions of organizational patterns.

Modeling textual descriptions of organizational patterns comes with one important limitation: multilayer perceptrons cannot learn global tendencies. Multilayer perceptrons can only be used to encode local tendencies and do not embrace global relationships between the frequencies of terms used to document organizational patterns in the sense of term frequency–inverse document frequency (TF-IDF).

Because consecutively applying affine transformations in artificial neurons still leads only to another affine transformation, activation functions of the rectified linear unit type from Hinton [20] had to be used on top of hidden layers to produce the information exchanged between hidden layers. A rectified linear unit is used as an activation function in exchange for its counterpart sigmoid activation function. Using a rectified linear unit as an activation function pays off when one cannot be sure that weights have been properly initialized at the start of the learning process. This is its primary advantage against the use of the sigmoid activation function. If the output from the sigmoid activation function comes close to 0 or 1, its gradient is close to zero, and it stops the learning process. This situation is characterized as a vanishing gradient that stops the learning process because the weights can no longer be updated (the weights cannot be updated because their update is zero).

5. Implementation

The mini-batch stochastic gradient descent method by Zhang et al. [16] was used in training the neural network. Because it is a gradient-based method capable of performing automatic differentiation, the Keras application programming interface of the TensorFlow platform was chosen for model learning. The code of the artificial neural network was written in Python programming language, which was subsequently translated into C programming language to speed up arithmetic operations. This translation helps the TensorFlow framework to efficiently self-manage computer memory.

The mini-batch stochastic gradient descent method was chosen as the most widely used method to train artificial neural networks. The Adam optimizer [21] was applied to train and enhance parameters by a multilayer perceptron. This algorithm allows a perceptron to optimize even when some hidden layers in back-propagation produce no information to be learned from. It was also shown that optimizers such as Adam with proven properties behave as if batch normalization is applied. Model parameters were optimized in 100 epochs in trigrams, finishing with a model with 2809 parameters. The number of epochs here is a hyper-parameter of the model. These were the parameters needed for the model assumed to be capable of computing probabilities that fragments of textual descriptions can be ascribed to the 19 selected organizational patterns documented by Coplien and Harrison [3].

The model overfit can be ascribed to the fact it worked with a very tiny dataset where the number of attributes was almost equal to the number of data instances. This dataset should be extended in future work with frequencies for more than triplets of words for organizational patterns documented in other sources. The created model was found to be sensitive to the similarity of inverse frequencies of terms from textual descriptions of organizational patterns. This means that if frequencies of these terms are similar, the model loses its ability to make accurate predictions of terms to organizational patterns. According to Bishop [22], a simpler model capable of generalizing well and achieving at least a bit lower error on the training dataset than an error on the validation or test dataset is generally preferred. According to Bishop [22] and Srivastava [23], dropout layers used in our experiments help multilayer perceptron be less sensitive to changes and measurement errors in input data to the neural network.

The overfit of the softmax regression model by Zhang et al. on the training dataset was partially solved by applying the L_1 regularization technique with dropout layers. Using dropout layers meant adding another type of hyper-parameter that determines the behavior of the artificial neural network. The L_1 regularization is a technique that adds a penalty term to the loss of each of the predictions (e.g., by computing the Frobenius norm of weight matrices presented by Zhang et al. [16]). Thus, the optimization algorithm helps the model minimize overly high parameter values. One of the advantages of the L_1 regularization is that, on average, it primarily strives to pay attention to the most important attributes of organizational patterns in the dataset, and those less important are learned less or with less-valued weights.

6. Experiments

To evaluate the method proposed in previous sections, we conducted four experiments on selected organizational patterns documented by Coplien and Harrison [3]. Two experiments were conducted with pairs of words, i.e., bigrams, and two with triplets of words, i.e., trigrams. The neural network architecture for the experiments with bigrams included more layers than those with trigrams, together with specialized regularization layers to overcome model overfitting of the training data. Two pattern sequences were established in each experiment. For each pattern sequence, a pattern story was created in order to check its meaningfulness. All four experiments, along with established pattern sequences and corresponding patterns stories, are available in a GitHub repository.¹ Due to the lack of space, we present only selected results here.

In the first experiment with trigrams, the Apprenticeship → Domain Expertise In Roles → Stand Up Meeting pattern sequence was established by running the model on the validation set, while the Architecture Team → Few Roles → Generics and Specifics was established by running the model on the test set.

In the second experiment with trigrams, the Organization Follows Market → Developing In Pairs → Architecture Team pattern sequence was established by running the model on the validation set, while the Smoke Filled Room → Code Ownership → Lock Em Up Together pattern sequence was established by running the model on the test set.

Only the first three highest probabilities in the model outputs were considered, so all pattern sequences established in the experiments consist of three organizational patterns. The method itself does not limit the length of pattern sequences. Because neural networks were trained on different values of parameters in each run, it was possible to establish unique pattern sequences.

Although the model is a deep neural network and should be strong enough, it cannot classify most organizational patterns based on the triplets of terms used to document them. The model in the first experiment worked accurately with trigrams and identified the Apprenticeship and Organization Follows Market organizational pattern in the validation set and Size the Organization in the test set. The obvious incapability of the model to classify trigram frequencies is probably a consequence of similar trigram frequencies in the dataset.

Let's look at the Apprenticeship → Domain Expertise In Roles → Stand Up Meeting pattern sequence. This is how it was established:

1. The probability of using Apprenticeship in the pattern sequence was 0.998081. This organizational pattern is about having the senior mentor with a junior software developer who teaches him the advanced skills needed for the job.
2. The probability of using Domain Expertise In Roles in the pattern sequence was 0.063485. This organizational pattern is about concentrating developers around domain experts or subject matter experts who lead the development and maintenance of the products.
3. The probability of using Stand Up Meeting in the pattern sequence was 0.061498. This organizational pattern is about meeting the development team at frequent and stable intervals, discussing problems, topics, and the next work plan.

¹<https://github.com/viktorFIIT/fiit-research-resources/tree/main/neural-network>

This pattern sequence documents the usual setting in development teams created by Size the Organization organizational pattern from Coplien and Harrison [3]. Novice developers work together with senior developers who are experts in their field. The development team usually consists of the subteams responsible for particular products, and these subteams must meet to discuss what's going on and work plans for the future. As we can see, this pattern sequence can be described in a pattern story, so we can consider it expected.

Let's look at the Architecture Team → Few Roles → Generics and Specifics pattern sequence. This is how it was established:

1. The probability of using the Architecture Team in the pattern sequence was 0.057077. This organizational pattern involves creating a small team to define the initial architecture.
2. The probability of using Few Roles in the pattern sequence was 0.055865. This organizational pattern is about having only a small number of producer roles in the team.
3. The probability of using Generics and Specifics in the pattern sequence was 0.055788. This organizational pattern is about letting experienced senior developers create the framework, let novices use it, and include it in specific applications.

This pattern sequence documents the usual setting of working on the development project. First, experienced senior developers choose the technology to work with, and then they design the initial software product architecture. Some less experienced software developers are then chosen to integrate and customize this framework in specific software products. Again, this pattern sequence can be captured in a pattern story, so we can consider it expected.

7. Discussion

The method proposed in this article fails to handle situations when a pattern textual description contains links to other patterns that should not or must not be used after this one. The first experiment with the trigrams network achieved similar classification accuracy in training and test datasets. In the first experiment, two organizational patterns were recognized in validation and one organizational pattern in the test dataset. The classification accuracy of working with the validation set fell from 0.111 to 0.11 after adding one duplicate record to the dataset. Adding one duplicate did not impact the network's ability to recognize 2 out of 18 organizational patterns.

According to Sousa et al. [11], professionals should assess how likely pattern sequences will be adopted. This means that the community must validate the value of expected sequences.

Documented patterns can be trusted to be true patterns no matter how often they occur. Kohls et al. [24] count uses of patterns before evaluating the true existence of patterns.

Souse et al. [11] note that very few authors assess the implementation of patterns with practitioners. This can lead to avoiding patterns that may be conditionally usable because their use in sequences depends on the technology used for their implementation.

According to Zhang et al. [16], the asymptotic standard deviation of the empirical error of the population error is $\sqrt{0.25/19}$. If there is a need to have 95% confidence and this difference is in an interval ± 0.01 using asymptotic analysis, then at least 10 000 rows like those in Table 1 must be collected in a dataset. Test sets must not be reused during repeated experiments, no matter whether subsequent runs of the model provide new examples of pattern sequences.

Because the number of produced pattern sequences Q can be dramatically higher than the number of inputs (19 in the experiments conducted in this article), the computational cost $\mathcal{O}(19 * Q)$ can be prohibitively high using 10 000 rows in the dataset.

According to Zhang et al. [16], accuracy can be traded off for computational and storage cost of the neural network and a quaternion-like decomposition can be used to decrease computational cost $\mathcal{O}(19 * Q)$ by the factor of n into the computational cost of $\mathcal{O}((19 * Q)/n)$.

8. Related Work

Waseeb et al. [13] proposed an automatic approach to discover relationships between patterns using text mining and natural language processing techniques. A similar dataset is used in this article, but the existence of n-grams in the descriptions of organizational patterns is replaced with term frequencies of bigrams and trigrams. Mining relationships between patterns from their textual descriptions needs to understand the meaning of the text, not just pattern names mentioned within the pattern textual descriptions.

Zhang et al. [16] presents sequence models, which, if implemented as recurrent neural networks, can be used to calculate the likelihood of use of pattern sequences, even if these are outputs from language models. According to Zhang et al. [16], sequence models can be optimized for producing the most likely used pattern sequences. The disadvantage of using Markov models to model pattern sequences is that patterns used long ago in the past are not considered in producing long pattern sequences.

Zanoni et al. [25] present an approach to predict the next design pattern to be used. Their approach requires a system architecture because it is based on extracting its features. They use a graph model to compare the structure of the existing system with a structure proposed by the design pattern.

Wijerathna et al. [26] proposed the first automated approach to learn text features from a corpus of 66 000 textual descriptions of software design problems from the Stack Overflow website through neural embeddings to predict the most likely design pattern to use to solve them. Compared to the method proposed in this article, they evaluated their method using accuracy, precision metrics, and through different models with different sets of hyper-parameters. Unlike the method proposed in this article, their approach is based on unsupervised learning. It requires a small number of labels compared to the dataset size used in this article to achieve very high accuracy (82%).

Building organizations with organizational patterns this article deals with is not unrelated to common project management problems [27, 28, 29, 30, 31], but it tends to happen in a more natural way.

9. Conclusions and Further Work

Explicit relationships included in textual description of patterns are insufficient to establish pattern sequences. This article brings a new method of establishing pattern sequences using artificial neural networks. The method is based on extracting implicit relationships between organizational patterns using the softmax regression model by Zhang et al. [16], which are

learned by the artificial neural network and encoded in its weights. The implementation of the method is based on the mini-batch stochastic gradient descent method by Zhang et al. [16] and the Keras application programming interface of the TensorFlow framework.

To evaluate the method, we conducted four experiments on selected organizational patterns documented by Coplien and Harrison [3]. Two pattern sequences were established in each experiment. For each pattern sequence, a pattern story was created in order to check its meaningfulness.

We intend to experiment further with the method and to apply it to a recently published catalog of security patterns [32].

Acknowledgments

The work reported here received funding from the Slovak national project Increasing Slovakia's Resilience Against Hybrid Threats by Strengthening Public Administration Capacities (Zvýšenie odolnosti Slovenska voči hybridným hrozbám pomocou posilnenia kapacít verejnej správy) (ITMS code: 314011CDW7) and from the Operational Programme Integrated Infrastructure for the project: Support of Research Activities of Excellence Laboratories STU in Bratislava (ITMS code: 313021BXZ1), co-funded by the European Regional Development Fund (ERDF).

References

- [1] C. Alexander, *The Timeless Way of Building*, Oxford University Press, 1979.
- [2] C. Alexander, S. Ishikawa, M. Silverstein, *A Pattern Language: Towns, Buildings, Construction*, Oxford University Press, 1977.
- [3] J. O. Coplien, N. B. Harrison, *Organizational Patterns of Agile Software Development*, Prentice-Hall, 2004.
- [4] L. Xu, J. Wang, X. Fang, Y. Zheng, D. He, Research on experimental teaching patterns based on e-learning, in: *Proceedings of 2008 International Conference on Computer Science and Software Engineering, CSSE 2008*, 2008.
- [5] V. Vranić, A. Vranić, Drama patterns: Extracting and reusing the essence of drama, in: *Proceedings of 24th European Conference on Pattern Languages of Programs, EuroPLoP 2019*, ACM, Irsee, Germany, 2019.
- [6] A. Vranić, V. Vranić, B. Vranić, Drama patterns: Seeing the patterns from within, in: *Proceedings of 24th European Conference on Pattern Languages of Programs, EuroPLoP 2019*, ACM, Irsee, Germany, 2019.
- [7] V. Vranić, A. Vranić, W. S. Khail, Growing organizations with patterns: Lessons from drama, in: *Proceedings of 25th European Conference on Pattern Languages of Programs, EuroPLoP 2020 Online*, ACM, 2020.
- [8] A. Vranić, V. Vranić, B. Vranić, Dramatizing software patterns: Focus group report, in: *Proceedings of 28th European Conference on Pattern Languages of Programs, EuroPLoP 2023*, ACM, Irsee, Germany, 2023.
- [9] B. Vranić, V. Vranić, Patterns of recreating reality in games, in: *Proceedings of 29th Conference on Pattern Languages of Programs, PLoP 2022*, ACM, 2022.

- [10] N. Seidel, Empirical evaluation methods for pattern languages: Sketches, classification, and network analysis, in: Proceedings of 22nd European Conference on Pattern Languages of Programs, EuroPLoP 2017, ACM, Irsee, Germany, 2017.
- [11] T. B. Sousa, H. S. Ferreira, F. F. Correia, A survey on the adoption of patterns for engineering software for the cloud, *IEEE Transactions on Software Engineering* 48 (2022) 2128–2140.
- [12] M. Falkenthal, U. Breitenbücher, F. Leymann, The nature of pattern languages, 2018.
- [13] S. Waseeb, W. S. Khail, H. G. Wahaj, V. Vranić, Extracting relations between organizational patterns using association mining, in: Proceedings of European Conference on Pattern Languages of Programs 2020, EuroPLoP 2020, ACM, Virtual Event, Germany, 2020.
- [14] R. K. Kaliyar, A multi-layer bidirectional transformer encoder for pre-trained word embedding: A survey of BERT, in: Proceedings of 10th International Conference on Cloud Computing, Data Science 'I&' Engineering, Confluence 2020, Noida, India, 2020.
- [15] B. Yu, C. Deng, L. Bu, Policy text classification algorithm based on Bert, in: Proceedings of 11th International Conference of Information and Communication Technology, ICTech 2022, Wuhan, China, 2022, pp. 488–491.
- [16] A. Zhang, Z. C. Lipton, M. Li, A. J. Smola, Dive into deep learning, arXiv preprint arXiv:2106.11342, <https://arxiv.org/abs/2106.11342>, 2021.
- [17] L. V. Jospin, H. Laga, F. Boussaid, W. Buntine, M. Bennamoun, Hands-on Bayesian neural networks—a tutorial for deep learning users, *IEEE Computational Intelligence Magazine* 17 (2022) 29–48.
- [18] J. S. Bridle, Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters, in: Proceedings of the 2nd International Conference on Neural Information Processing Systems, MIT Press, 1989.
- [19] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, arXiv preprint arXiv:1502.03167, <https://arxiv.org/abs/1502.03167>, 2015.
- [20] V. Nair, G. E. Hinton, Rectified linear units improve restricted Boltzmann machines, in: Proceedings of 27th International Conference on Machine Learning, ICML 2010, Haifa, Israel, 2010, p. 807–814.
- [21] D. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Proceedings of 3rd International Conference for Learning Representations, ICLR 2015, San Diego, USA, 2015.
- [22] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.
- [23] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, *Journal of Machine Learning Research* 15 (2014) 1929–1958.
- [24] C. Kohls, S. Panke, Is that true...? thoughts on the epistemology of patterns, in: Proceedings of 16th Conference on Pattern Languages of Programs, PLoP 2009, Chicago, Illinois, USA, 2009.
- [25] M. Zanoni, F. Arcelli Fontana, F. Stella, On applying machine learning techniques for design pattern detection, *Journal of Systems and Software* 103 (2015) 102–117.
- [26] L. Wijerathna, A. Aleti, Predicting software design patterns from text using neural embedding, in: Proceedings of 35th IEEE/ACM International Conference on Automated Software Engineering, ASE '20, Virtual Event, Australia, 2021.
- [27] I. Teslia, I. Khlevna, N. Yehorchenkova, O. Grigor, Y. Kataieva, T. Latysheva, T. Prokopenko,

- Y. Tryus, A. Khlevnyi, Development of the concept of building project management systems in the context of digital transformation of project-oriented companies, *Eastern-European Journal of Enterprise Technologies* 6 (2022) 14–25.
- [28] I. Teslia, N. Yehorchenkova, O. Yehorchenkov, I. Khlevna, Y. Kataieva, V. Veretelnyk, B. Khmelnytsky, I. Chastokolenko, I. Ohirko, A. Khlevnyi, T. Shevchenko, T. Latysheva, Development of the concept of construction of the project management information standard on the basis of the Primadoc information management system, *Eastern-European Journal of Enterprise Technologies* 1 (2022) 53–65.
- [29] I. Teslia, N. Yehorchenkova, O. Yehorchenkov, H. Zaspas, I. Khlevna, Development of principles and method of electronic project management, *Eastern-European Journal of Enterprise Technologies* 5 (2017) 23–29.
- [30] N. Yehorchenkova, O. Yehorchenkov, Y. Kataieva, S. Mitsenko, O. Mohylnyi, S. Odokienko, N. Babina, O. Verenysh, Models and methods of project administration in 4P environment, in: *Proceedings of 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2021*, IEEE, Krakow, Poland, 2021.
- [31] N. Yehorchenkova, I. Teslia, O. Yehorchenkov, L. Kubiavka, T. Latysheva, Y. Kataieva, V. Olena, Model of management of resources production in 4P-environment of project-oriented enterprise, in: *Proceedings of 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2020*, IEEE, Dortmund, Germany, 2020.
- [32] A. Cordeiro, A. Vasconcelos, M. Correia, A catalog of security patterns, in: *Proceedings of 29th Conference on Pattern Languages of Programs, PLoP 2022*, ACM, 2022.