Genetic algorithm based sentence packaging in natural language text generation

IOP Conf. Series: Materials Science and Engineering **537** (2019) 042003 doi:10.1088/1757-899X/537/4/042003

Genetic algorithm based sentence packaging in natural language text generation

Dmitry Devyatkin, Vadim Isakov* and Alexander Shvets

Federal Research Center «Computer Science and Control» of Russian Academy of Sciences, Moscow, Russian Federation

* E-mail: isakov@isa.ru

Abstract. Sentence packaging is an important task in natural language text generation which could be treated as a particular kind of a community detection problem. We propose an approach based on genetic algorithm and predictive machine learning models to advance it. The approach allows handling large ontological and semantic structures in a form of a graph to produce well-formed sentences. The results of experiments showed that the genetic algorithm optimizing the modularity measure gives comparable results to ones achieved by a traditional community detection algorithm and outperforms it on a collection of relatively short texts. The design of an approach allows for further introducing linguistic characteristics into a fitness function that gives it a high potential to increase the quality of detected packages while taking into account the specificity of the domain.

1. Introduction

One of the main tasks in natural language text generation (NLTG) consists in converting structured non-linguistic data kept in a form of a graph (i.e. ontological or semantic structures [1-4], e.g. RDF-graphs which combine single RDF-triples with cross-referenced elements) to a coherent text. The following basic steps in the generation process are generally highlighted [5]:

- Content determination, i.e. selecting the part of the data in bits of information which should be verbalized with a text.
- Text structuring, i.e. determination of the order in which information should appear in the text.
- Sentence aggregation, i.e. combining multiple messages into larger sentences to have potentially more fluid and readable text.
- Lexicalization, i.e. choosing an appropriate lexis for the sentences.
- Referring expression generation, i.e. selecting word or phrases to communicate sufficient information to distinguish one domain entity from other domain entities.
- Linguistic realization, i.e. forming a well-formed sentence.

The mentioned steps are often considered separately. The approaches handling these steps are usually based on a combination of hand-crafted rules and machine learning models. Pure rule-based methods were mostly investigated in early works in NLTG devoted to English as an analytic language. For example, in [6], a rule-based method was offered for generating natural language texts from business process models. Recently, with developing the machine learning techniques, fusional

languages, such as Russian or Spanish, began to be covered in studies more often. In [7] an inflection module to improve the naturalness and expressivity of obtained texts in Spanish on the lexicalization stage was offered. This module inflects the verbs using an ensemble of machine learning models. The other types of words are inflected using hand-crafted rules.

Some neural network architectures allow creating the whole end-to-end NLTG frameworks in an unsupervised manner skipping some common steps in the generation and still achieve a relatively high quality of generated texts. Li et al. [8] proposed a hierarchical LSTM autoencoder that generates a latent representation for a text from embeddings for sentences and words and then uses this embedding to restore the original text. They also empirically showed that neural models are able to encode texts in such a way that preserve syntactic, semantic, and discourse coherence. In [9] researchers proposed a deep ensemble of autoencoders with attention layer for reports generation. They introduced the semantic reranking approach to infer the output of the ensemble. Variational autoencoders are the most promising approach in this direction because they allow including a priori linguistic information in the model; cf., e.g., [10] describing hierarchical latent variable recurrent network or [11] proposes a hybrid variational autoencoder architecture that includes convolutional, deconvolutional, and recurrent components.

Despite the recent breakthrough in separate subtasks in NLTG, it becomes clear that neither combining the detached approaches which cover just single generating stages nor end-to-end frameworks ignoring some stages allow reaching all the goals set in a traditional generation pipeline to obtain satisfactory text. Therefore, a tendency toward carrying out at least some steps simultaneously is being established [12-14].

In this paper, we tackle a task called sentence packaging [15] that consists in decomposing the initial graph containing the data to be verbalized into connected subgraphs which would include all the information sufficient for translating them into well-formed sentences. This step partially touches content determination, text structuring, and sentence aggregation avoiding unnecessary predefining the elementary statements with a subsequent positioning and grouping. Many criteria apply to the design of packages including language, style, topic, the presence of the obligatory arguments of predicative vertices, linguistic restrictions of co-reference (in case several subgraphs share the same vertices), etc. In the current work, we focus only on the formal criterion implying that each package should be in itself a connected graph. The rest of the criteria are taken into account implicitly while trying to recover the ground truth sentences which satisfy them a priori.

In graph-theoretical terms, sentence packaging can be considered as an approximation of dense subgraph decomposition, which is a very prominent area of research in graph theory. It has been studied in the context of numerous applications, including biomedicine [16], web mining [17], influence analysis [18], acoustic source separation with use of deep clustering approaches [19], community detection [20], etc. The latter application and corresponding methods for extracting communities are of particular interest for the task since they are often designed to work with extremely large graphs and some of them may take into account additional features of nodes. Some methods could separate subgraphs even if they overlap, that is an often case in the generation when there is a need to duplicate some nodes and split a complex sentence into several smaller ones in order to have a coherent text.

The traditional community detection methods [21, 22] are based on optimizing some measure related to the network structure by surveying a graph and grouping the nodes in such a way that subsequently increases the objective measure. It was showed previously that these methods are applicable to sentence packaging and give promising results [23]. Further, to improve the quality of packages some additional to network structure features should be taken into account. They may include linguistic characteristics of nodes, parental relations, type of relations, type of arguments and possible combination of them in a sentence. The objective function mainly reflected the density of nodes inside and between communities should be changed regarding new linguistic features. However, the modifying the function is not enough since there is no guarantee the design of algorithms will

allow them to investigate the search space in the same efficient way they did being tuned for optimizing network measure.

To overcome the mentioned problem and to avoid adapting existing algorithms for each particular objective function being developed while looking for an effective one, the evolutionary algorithms (EA) might be chosen as an appropriate approach for a detection of different communities with a specific structure and distinct properties. The main advantage of EA for the sentence packaging task consists in a possibility to carry out the search of suboptimal solutions without the necessity to have knowledge about the search space that depends on the function to be optimized. Moreover, genetic algorithm (GA) in particular has been already used in several works on community detection [24, 25] showing a competing performance in comparison to traditional approaches.

The aim of this work consisted in checking whether it is possible to apply a genetic algorithm to carry out sentence packaging. We show that using a predicted number of packages and the network modularity measure [26] as a fitness function allows obtaining sentences highly corresponding to the ground truth ones. That gives us an opportunity to customize the objective function by introducing linguistic information in order to improve the quality of packaging. The remainder of the paper is structured as follows. In section 2, we describe the developed method for graph decomposition. Section 3 provides the data for English and Russian languages for training the models and testing the method. Section 4 covers experiments we carried out. In Section 5, we provide the results of experiments. Section 6 is devoted to the discussion of the outcomes of the work. In Section 7, we draw some conclusions and suggest future directions of research.

2. Method

We propose a method that consists of the combination of a linear regression model and a genetic algorithm. The linear regression model is trained to predict the approximate number of packages for a given graph while the genetic algorithm takes this number to reduce the search space and decomposes a graph into densely connected subgraphs. To predict the most probable number of sentences the graph should be decomposed into, we used different linguistic features within a ridge regression model similar to the work [23]. For Russian, we considered frequencies of the following elements of the graphs as features for the prediction: tokens, predicates, arguments, syntactic roots, syntaxemes [27], subjects, and objects. For English, the following features were used: tokens, edges, predicates, arguments, roots, VerbNet nodes, and different relation types from PennTreebank (Argument1, Argument2, Elaboration, NonCore, and Set) [28]. The predicted number of sentences might optionally influence the initialization phase of GA. There is a lot of works in evolutionary computation for community detection which often offer to handle the task as multiobjective optimization, suggest different genotype encoding schemes (some of them allow detecting overlapping communities), different crossover and mutation operators, and some additional modifications [29]. They have various advantages and drawbacks, however, we did not focus on them because the detected communities do not necessarily have the same quality if they are treated as sentence packages. Therefore, we prefer to have a version with a small number of parameters to easier check if there is a combination proper for the defined task.

We chose one of the first genetic algorithms designed for community detection [30] as a core algorithm for graph decomposition. It optimizes the network modularity measure proposed by Girvan and Newman [26] that has been already successfully used for sentence packaging in [23]. The measure is a metric for assessment of partitioning a network into communities which is defined as:

$$Q = \sum_i e_{ii} - a_i^2 \tag{1}$$

where i is the index of the communities, e_{ii} is the fraction of edges that connect vertices inside the community i (i.e., within-community edges) and a_i is the fraction of the edges that connect a vertex to vertices in community i. If the number of within-community edges is no better than random, $Q = 0$. Values approaching $Q = 1$, which is the maximum, indicate strong community structure.

The first reason for selecting the algorithm is the highly frequent convergence to the correct solution on classical tasks in community detection shown in [26]. The second reason is a relatively small number of parameters needed to be tuned. The third reason is the ability of the algorithm to handle large graphs (at least up to 100,000 nodes) that allows planning to introduce complex objective functions which might take a long time to be computed. Moreover, it does not require the exact number of communities as an input that is also an advantage because the number of sentences to be used for verbalization of the data is unknown.

The original algorithm in [26] has several peculiarities in comparison to the standard GA. First of all, a genotype is represented as an array of positive natural numbers which assign each node of the graph to some community with the corresponding identifier. Random initialization of population is extended by taking several nodes (the percentage of nodes to be taken is an additional parameter of the algorithm that should be tuned) and assigning their identifiers to their neighbors. Another distinction is in the implementation of cross-over operation. The nodes from a source chromosome corresponding to the same randomly selected community identifier are transferred to the destination chromosome. The mutation is performed by placing a node into a random community. Additional operation called "clean-up" consists in changing the identifier of a node and its neighbors to the identifier belonging to the majority of neighbors in case the relevant number of nodes forming the majority is higher than some predefined threshold (the threshold and the number of nodes to be taken for cleaning are two other new parameters requiring tuning).

Since authors of the original algorithm have not provided preferable values for new parameters, we ran several experiments and defined the values based on empirical observations as follows: the percentage of nodes to be taken during initialization equals 70%, the number of nodes to be considered during cleaning phase is equal to 30% with a threshold for the majority of neighbors belonging to the same community equal to 70%. We selected tournament selection operator and used the same measure, network modularity, as an objective function.

Although the algorithm could have been run "as is", the following questions related to the peculiarities of the domain arose: a) whether the algorithm has to be applied to the whole graph of the text or to each connected subgraph separately (it was shown previously [23] that some traditional graph decomposition algorithms work better being applied to the whole graph of text rather than to its isolated subgraphs); b) whether we should limit the possible number of packages by reducing the number of identifiers used for generating the initial population or leave the decision of how many sentences to obtain fully to the genetic algorithm; c) whether we should return obtained sentences even if some of them are in a form of disconnected subgraphs or treat such subgraphs as individual sentences. To answer these questions, we introduced three additional parameters to the algorithm: a flag respondent for the preliminary splitting of the graph, maximum possible number of community identifiers to be used for initialization, and a flag respondent for splitting resulting sentences into isolated subgraphs. The second parameter might be either the total number of nodes or the most probable number of sentences predicted with the linear regression model. Regarding this, we created several instances of the algorithm with different values of parameters to find the proper combination of them.

3. Data

The graphs we utilized for experiments were built from linear texts by linking the tokens in syntactic and semantic relations and merging the tokens corresponding to the same entities inside and across sentences. The dataset for English is the one used in [23]. It is based on VerbNet/FrameNet annotated version of the Penn TreeBank [31], with identified and fused coreferences within the individual texts of the corpus with use of the Stanford CoreNLP toolkit [32]. To create a dataset for Russian, we took the annotated open corpus Rucor (http://rucoref.maimbava.net/) with anaphorical and coreferential relations between noun groups. Rucor contains 156,637 tokens and 3,638 coreference chains. The corpus consists of 181 texts and a corresponding table with lists of word offsets for each coreference chain. Part-of-speech tags, morphological annotation, dependency trees, and semantic roles were

extracted using natural language processor for Russian [33]. Figure 1 shows a curve located above the diagonal that indicates that the majority of sentences (73%) are connected. There are 5,962 connected sentences and 8,221 sentences in total with up to 150 connected sentences per text; cf. figure 2.

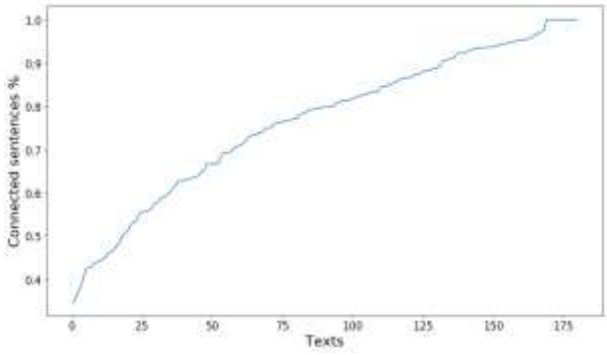

Figure 1. The percentage of connected sentences for each text.

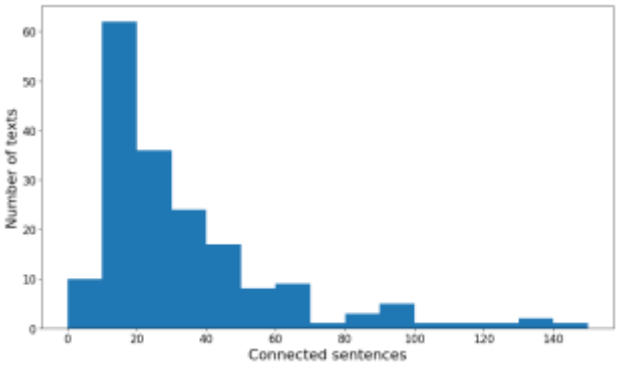

Figure 2. The number of connected sentences per text (histogram).

4. Experiments

The task was to decompose compound graphs into single packages as close as possible to initial ground truth sentences. We measured the quality of decomposition calculating recall, precision, and F1-score for each initial sentence and averaging the values. The dataset for each language was split into a development set and a test set in proportions 85% and 15%. The development set was used for training the linear regression model. The test set was used for a comparison of variants of the proposed algorithm which differ from each other by values of introduced parameters; cf. Section 5 for details.

To avoid the waste of computational time for the algorithms with an improper combination of values of parameters and to detect them faster, a collection of short texts was selected from the test set. The genotype corresponding to the largest text from this collection (in terms of the number of edges between tokens) has a length of 99. The predicted number of sentences for this text is 6, i.e. one of 6 community identifiers might be assigned to each gene. Despite the limitation on possible classes, the search area is still large to be fully surveyed. The successful algorithms were run then on the whole test set.

We did 100 independent runs for each algorithm on the collection of short texts and 25 runs on the whole test set. The significance of results was calculated using Mann-Whitney U-test [34].

5. Results

Regarding the first part of the approach, consisting in training linear regression model, we got the R^2-value up to 0.85 for Russian dataset and up to 0.97 for English one, thus the obtained models can be used for an accurate prediction of the number of sentences. Correlation between expected and predicted numbers of sentences in texts of the development sets for both languages are shown in figure 3. The values obtained within a test set by applying the trained model were used to limit the number of identifiers needed to generate the initial population in GA.

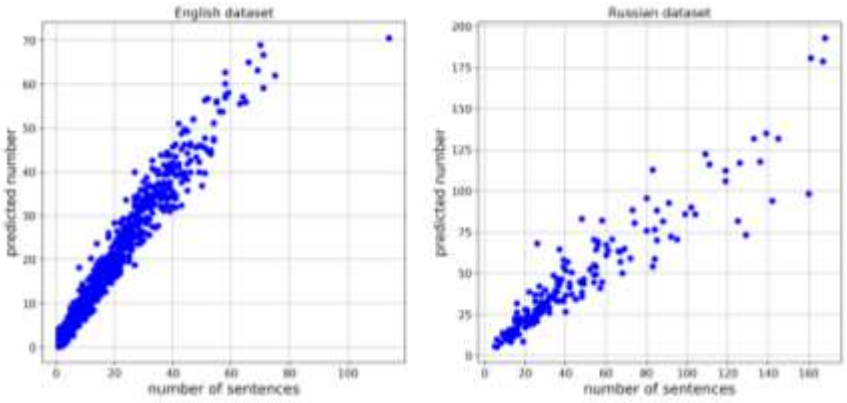

Figure 3. Predicting the number of sentences with linear regression on the development sets.

Results of different runs presented in tables 1-3, where "split/wo_split" stands for "with/without preliminary split to isolated subgraphs"; "non_fixed" stands for "taking the total number of nodes as number of possible identifiers", "predicted" stands for "the possible number of identifiers predicted with linear regression", and "fixed" stands for "exact ground truth number of sentences (possible identifiers)". The numbers in the title of an algorithm correspond to the number of individuals in a population and to the number of generations.

LOUVAIN [21] algorithm widely used for community detection was chosen as a baseline since it was designed for the fast unfolding of communities in large networks. The algorithm consists in a hierarchical optimization of the same modularity measure we use in our work and includes two main phases that are repeated iteratively: finding a local maximum of the modularity and building a new network whose nodes are then the communities found during the first phase. "No decomposition", in table 3, corresponds to the simple splitting of initial graphs into isolated subgraphs and treating them as the resulting sentences.

Table 1. Results of performing the algorithms on short texts (non-fixed/predicted/fixed, split/wo_split).

Algorithm	Recall	Precision	F_1-score
LOUVAIN	0.65	0.95	0.73
GA200x20_wo_split_non_fixed	0.73	0.86	0.76
GA200x20_split_non_fixed	0.58	0.97	0.67
GA200x20_wo_split_predicted	0.73	0.88	0.77
GA200x20_split_predicted	0.83	0.87	**0.83**
GA500x40_wo_split_non_fixed	0.67	0.93	0.75
GA500x40_split_non_fixed	0.58	0.97	0.68
GA500x40_wo_split_predicted	0.78	0.86	0.80
GA500x40_split_predicted	0.83	0.88	**0.84**
GA500x40_split_fixed	0.89	0.91	**0.89**

Table 2. Results of performing the algorithms on short texts with different resources.

Algorithm	Recall	Precision	F_1-score
GA500x40_split_predicted	0.831	0.877	**0.838**
GA500x20_split_predicted	0.821	0.880	0.830
GA300x30_split_predicted	0.831	0.876	0.837
GA500x100_split_predicted	**0.832**	0.877	**0.838**

Table 3. Results of performing the algorithms on all English texts from the test set.

Algorithm	Recall	Precision	F_1-score

No decomposition	0.31	0.27	0.28
LOUVAIN	0.66	0.66	**0.64**
GA200x20_wo_split_predicted	0.43	0.90	0.52
GA200x20_split_predicted	0.56	0.88	0.62
GA500x40_split_predicted	0.60	0.85	**0.64**

We conducted the Mann-Whitney U-test to obtain statistical significance of the difference between LOUVAIN and GA500x40_split_predicted F_1-scores. The results showed that GA500x40_split_predicted algorithm has outperformed LOUVAIN on the short texts with z-score = 12.22 (p-value < $1.8*10^{-35}$) and on the whole test set with z-score = 4.92 (p-value < $4.6*10^{-07}$). The GA500x20_split_predicted algorithm is the only algorithm in table 2 that differs from the others with a statistical significance (each p-value < $3.4*10^{-08}$).

6. Discussion

Table 1 shows that with the resources 200x20, the algorithm does not converge often close enough to the solution and therefore it is difficult to compare the performance of instances with different parameters. With more resources, the difference between algorithms is clearer. However, regardless of the resources, it is seen that the best solution is achieved when the graph is split and the approximate predicted number of sentences is used. The last row in table 1 shows the values which might be achieved within given resources if we use ground truth number of sentences.

We also tried to return the final sentences in two ways: as they were stored in the best genotype and selecting isolated connected components from each sentence before returning them that might have increased the number of final sentences. In the first case, we had the average recall, precision, and F_1-score equal to (0.82, 0.85, 0.82), in the second case, they were a bit higher – (0.82, 0.89, 0.83). If we return only the largest connected component for each detected sentence we get low scores - (0.8, 0.86, 0.81). Thus, additional decomposition of final communities into parts in case they consist of separate connected components gives better F_1-score. However, including this step to the evaluation of fitness function gives worse results, therefore this step should be performed only once after the termination of the algorithm.

Considering table 2, we can see that increasing the number of resources (the size of the population and the number of generations) leads to better results. It means that modularity is an appropriate measure to form the basis of an objective function. At the same time, the Mann-Whitney U-test shows that the difference is not significant after a certain point. It means that to achieve higher values of F_1-score the function should be improved.

Splitting the graph into several isolated subgraphs and applying sentence packaging to each subgraph independently gives higher F_1-score rather than applying it to the whole graph. That distinguishes the algorithm from LOUVAIN which gives better results when applied to the whole graph. Thus, determining the approximate number of sentences with linear regression and therefore reducing the number of possible solutions improves the convergence of the algorithm. Moreover, an opportunity of the algorithm to decrease naturally the number of community identifiers eliminates the necessity of running it several times with different predefined numbers of packages.

Table 3 demonstrates that GA500x40_split_predicted algorithm performs similarly to the LOUVAIN algorithm in terms of F_1-score achieving higher precision. The main advantage of a proposed approach over the baseline algorithm is that the optimized measure (the network modularity) might be accompanied by the domain-specific information without the necessity to change the design of an algorithm.

7. Conclusions and future work

We have presented an approach to performing the sentence packaging in natural language text generation which comprises predictive machine learning models and specifically designed genetic algorithm to tackle the task as a community detection problem.

We showed that the modularity as a measure to be optimized with GA is a proper basis for the task. The approach allows decomposing large graphs into a set of dense subgraphs without the necessity to predefine the number of communities. At the same time, there is a possibility to limit this number that in practice, according to the results of experiments, leads to increasing the quality of formed packages in terms of F_1-score.

We have also created a suitable for the task dataset for the Russian language within this work and enumerated linguistic and network features both for English and Russian which give the highest R^2-value being used in linear regression for predicting the probable number of sentences in a particular graph.

The next prompting step is to integrate linguistic information into the objective function in order to increase the quality of generated sentences. The objective function might be based on the closeness of a sentence to the multivariate normal distribution (MVN) of different linguistic characteristics, similar to the ones used for improving obtained packages by the descent search in [23]: number of tokens, number of edges, number of predicate nodes, number of argument nodes, number of roots (vertices without input edges), number of verbal and nominal predicate tokens, number of arguments of each type, the type(s) of the parent node(s) of each node and the types of its arguments. The MVN distribution might be trained using the development set. In case, the development set is preliminarily clustered, the joint distribution might be built separately for each cluster and then the highest degree of correspondence to given MVN distributions might be chosen as a value to be maximized. The composition of this value with the network modularity value will form the final objective function.

A multi-objective genetic algorithm could be a promising approach for simultaneous optimization of different measures that will give an opportunity to find a trade-off between sets of packages with proper network structure and with consistent linguistic characteristics. Combining it with the co-evolutionary scheme will allow avoiding tuning the own parameters of GA.

Acknowledgments
The presented work was supported by the Russian Foundation for Basic Research under the contract number 18-37-00198.

References

[1] Franconi E, Gardent C, Juarez-Castro X and Perez-Beltrachini L 2014 Quelo natural language interface: Generating queries and answer descriptions *Natural Language Interfaces for Web of Data*
[2] Colin E, Gardent C, Mrabet Y, Narayan S and Perez-Beltrachini L 2016 The WebNLG challenge: Generating text from dbpedia data *Proc. of INLG* 163–7
[3] Bohnet B, Wanner L, Mille S and Burga A 2010 Broad coverage multilingual deep sentence generation with a stochastic multi-level realizer *Proc. of COLING* 98–106
[4] Flanigan J, Dyer C, Smith N A and Carbonell J 2016 Generation from abstract meaning representation using tree transducers *Proc. of NAACL:HLT* 731–9
[5] Gatt A and Krahmer E 2018 Survey of the State of the Art in Natural Language Generation: Core tasks, applications and evaluation *Journal of Artificial Intelligence Research* **61** 65-170
[6] Leopold H, Mendling J and Polyvyanyy A 2014 Supporting process model validation through natural language generation *IEEE Transactions on Software Engineering* **40(8)** 818-40
[7] Barros C, Gkatzia D and Lloret E 2017 Improving the Naturalness and Expressivity of Language Generation for Spanish *Proc. of the 10th International Conference on Natural Language Generation* 41-50
[8] Li J, Luong M T and Jurafsky D 2015 A hierarchical neural autoencoder for paragraphs and documents *arXiv* (*Preprint* arXiv:1506.01057)
[9] Juraska J, Karagiannis P, Bowden K K and Walker M A 2018 A Deep Ensemble Model with Slot Alignment for Sequence-to-Sequence Natural Language Generation *arXiv* (*Preprint* arXiv:1805.06553)

[10] Serban I V *et al* 2017 A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues *AAAI* 3295-301
[11] Semeniuta S, Severyn A and Barth E 2017 A hybrid convolutional variational autoencoder for text generation *arXiv* (*Preprint* arXiv:1702.02390)
[12] Konstas I and Lapata M 2012 Unsupervised concept-to-text generation with hypergraphs *Proc. of NAACL* 752–61
[13] Narayan S, Gardent C, Cohen S B and Shimorina A 2017 Split and rephrase *arXiv* (*Preprint* arXiv:1707.06971)
[14] Wen T H, Gaš´c M, Mrkš´c N, Su P H, Vandyke D and Young S 2015 Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems *Proc. of EMNLP* 1711–21
[15] Wanner L and François L 2009 Applying the Meaning-Text theory model to text synthesis with low-and middle-density languages in mind *Language Engineering for Lesser-Studied Languages* **21**
[16] Hagmann P, Cammoun L, Gigandet X, Meuli R, Honey C J, Wedeen V J and Sporns O 2008 Mapping the structural core of human cerebral cortex *PLoS Biology* **6(7)** 888–93
[17] Sarıyuece A E, Seshadhri C, Pinar A and Çatalyuek Ue V 2015 Finding the hierarchy of dense subgraphs using nucleus decompositions (*Preprint* arXiv:1411.3312v2)
[18] Ugander J, Backstrom L, Marlow C and Kleinberg J 2012 Structural diversity in social contagion *Proc. of the National Academy of Sciences* **109(16)** 5962–6
[19] Hershey J R et al. 2016 Deep clustering: Discriminative embeddings for segmentation and separation *Acoustics, Speech and Signal Processing (ICASSP) IEEE* 31-5
[20] Asim Y, Majeed A, Ghazal R, Raza B, Naeem W and Malik A K 2017 Community detection in networks using node attributes and modularity *Int J Adv Comput Sci Appl* **8(1)** 382–8
[21] Blondel V D, Guillaume J L, Lambiotte R and Lefebvre E 2008 Fast unfolding of communities in large networks *Journal of statistical mechanics: theory and experiment* **10** 10008
[22] Karypis G and Kumar V 2000 Multilevel k-way hypergraph partitioning *VLSI design* **11(3)** 285–300
[23] Shvets A, Mille S and Wanner L 2018 Sentence Packaging in Text Generation from Semantic Graphs as a Community Detection Problem *Proc. of the 11th International Conference on Natural Language Generation*
[24] Pizzuti C 2012 A multiobjective genetic algorithm to find communities in complex networks *IEEE Transactions on Evolutionary Computation* **16(3)** 418-30
[25] Cai Y, Shi C, Dong Y, Ke Q and Wu B 2011 A novel genetic algorithm for overlapping community detection *International conference on advanced data mining and applications* 97-108
[26] Newman M E J and Girvan M 2004 Finding and evaluating community structure in networks *Physical Review E* **69** 026113
[27] Osipov G et al. 2013 Relational-situational method for intelligent search and analysis of scientific publications *Proc. of the Integrating IR Technologies for Professional Search Workshop* 57-64
[28] Marcus M P, Marcinkiewicz M A and Santorini B 1993 Building a large annotated corpus of English: The Penn Treebank *Computational linguistics* **19(2)** 313-30
[29] Pizzuti C 2018 Evolutionary computation for community detection in networks: a review *IEEE Transactions on Evolutionary Computation* **22(3)** 464-83
[30] Tasgin M, Herdagdelen A and Bingol H 2007 Community detection in complex networks using genetic algorithms *arXiv* (*Preprint* arXiv:0711.0491)
[31] Kingsbury P, Palmer M and Marcus M 2003 Adding Semantic Annotation to the Penn TreeBank *Proc. of the human language technology conference*
[32] Manning C D, Surdeanu M, Bauer J, Finkel J, Bethard S J and McClosky D 2014 The Stanford CoreNLP Natural Language Processing Toolkit *Proc. of the 52nd Annual Meeting of the*

Association for Computational Linguistics: System Demonstrations 55–60
[33] Shelmanov A O and Smirnov I V 2014 Methods for semantic role labeling of Russian texts. *Computational Linguistics and Intellectual Technologies. Proc. of International Conference Dialog* **13(20)** 607-20
[34] Mann H B and Whitney D R 1947 On a test of whether one of two random variables is stochastically larger than the other *Annals of Mathematical Statistics* **18** 50-60