Generating Modern Poetry Automatically in Finnish

Mika Hämäläinen
Department of Digital Humanities
University of Helsinki
mika.hamalainen@helsinki.fi

Khalid Alnajjar
Department of Computer Science
University of Helsinki
khalid.alnajjar@helsinki.fi

Abstract

We present a novel approach for generating poetry automatically for the morphologically rich Finnish language by using a genetic algorithm. The approach improves the state of the art of the previous Finnish poem generators by introducing a higher degree of freedom in terms of structural creativity. Our approach is evaluated and described within the paradigm of computational creativity, where the fitness functions of the genetic algorithm are assimilated with the notion of aesthetics. The output is considered to be a poem 81.5% of the time by human evaluators.

1 Introduction

Poem generation is a challenging task for creative NLG (natural language generation) requiring structural integrity in the form of rhyming and meter, grammatical correctness and figurative expression. Poems are meant to be interpreted and therefore the meaning they convey cannot be fully explained by semantics, but they rather require an exploration into the notion of pragmatics.

In this paper, we present a novel approach based on a genetic algorithm for creating poetry in Finnish from the stand point of computational creativity. In addition to solving problems related to poems in general, the morphosyntactically complex Finnish sets additional requirements for producing grammatical output.

Computational creativity can be seen as a search for creative artefacts in a conceptual space (cf. Wiggins, 2006). Therefore the use of genetic algorithm for a creative task is reasonable as it conducts a search and picks out the most suitable candidates based on its fitness function. An important aspect for creativity is that the system should be able to assess its own creations, a notion called appreciation (Colton, 2008) or aesthetic function (Colton et al., 2011) in the literature. The fitness function of the genetic algorithm serves for this exact purpose, as it can score the output in terms of different aesthetic dimensions.

2 Related work

In the past, poetry generation has been studied both from the point of view of computational creativity and natural language generation. Poem generation has been tackled with a variety of different methods such as case-based reasoning (Gervás, 2001), templates (Colton et al., 2012), translation with WFSTs (weighted finite-state transducers) (Greene et al., 2010), text transformation via word embeddings (Bay et al., 2017) and conditional variational autoencoders (Li et al., 2018). As the field of poem generation has been broadly discussed by Oliveira (2017), we dedicate the rest of this section to describing the existing poetry generation work conducted for Finnish within the computational creativity paradigm. We also discuss some previous approaches using genetic algorithms.

One of the first takes on Finnish poem generation is the P. O. Eticus system (Toivanen et al., 2012). P. O. Eticus uses a corpus of human authored poems. These poems are used as templates for generating new poetry. In practice, the system takes a random poem from the corpus, conducts a morphological analysis on it and replaces some of the words in the existing poem. The replaced words are inflected to match the morphology of the original word.

Another take on the poetry generation in Finnish is that of Kantosalo et al. (2015). This approach is presented as a part of a poem authoring system. How this generator operates is that it takes sentences form children's books stored in its corpus based on a shared keyword. These sentences serve as verses, or poem fragments, and they are output one after another forming a generated poem. As the system does not alter text at all, it does not have to deal with the complexities of the Finnish morphology.

The most recent work on Finnish poem generation is the work presented by Hämäläinen (2018a). This approach consists of individual rule-based verse generators, each of which produces structurally different verses with different types of figurative expression, such as metaphors, tautology, comparison and so on. The verse generators are applied in the order defined by hand-written poem structures. Semantic cohesion is achieved by the fact that each verse generator outputs a noun to the following verse generator in the poem structure. This guarantees that verses are always coherent to some extent with the verse that immediately precedes them. This generator is in use in the creative internet application Poem Machine tailored for co-creativity (Hämäläinen, 2018b).

The previous approaches in Finnish poem generation covered in this section are limited in terms of structural creativity. The approaches are either limited by the structure imposed by the existing poems or sentences, or the hand-written verse structures. The approach we present in our paper showcases more creative freedom on the structural level. This, however, is challenging due to the complicated morphosyntax of Finnish; structural changes can easily render the results nonsensical as wrong morphology in an incorrect syntactic position will make the entire sentence nongrammatical. We take this into account in our proposed method.

Genetic algorithms have been used in the generation of poetic language before. Although not full poem generation, Hervás et al. (2007) have used genetic algorithms for generating alliterations in Spanish. In terms of full poetry generation, Manurung et al. (2012) aim for *grammaticality*, *meaningfulness* and *poeticness* with their genetic algorithm approach. Their approach tires to maximize the similarity of the poem meter to the target meter, and the poem semantics to the target semantics, while still retaining grammaticality.

A recent approach to poem generation with genetic algorithms, TwitSong 3.0 (Lamb and Brown, 2019), is based on a mined corpus of sentences that are used as verses in poems based on their inter-compatibility in terms of rhyming. They base their fitness functions on the following metrics: (*meter, emotion, topicality* and *imagery)*. The fitness functions operate on verse level. They solve emotion and imagery with existing lexicons, topicality is assessed based on trigram and keyword similarity with the desired topic and meter is scored based on how close it is to a iambic meter.

3 Poem Generator

This section is dedicated to describing the data used for poem generation, the genetic algorithm and how the Finnish morphosyntax is solved by the system. Special attention is paid to describing the fitness functions, according to which the system can rank its creations.

3.1 Data

We crawl Wikisources[1] for Finnish poetry. This way we obtain 6,189 poems. We parse the poems by using the Finnish dependency parser (Haverinen et al., 2014) to obtain syntactic relations, morphological features, part of speech and lemma for each word. This constitutes our poem corpus, denoted as P, with verse-level syntactic parsing. These poems are used by the genetic algorithm for the initial population, where a stanza of a human authored poem is treated as one poem.

For semantics, we use the pretrained word2vec word embeddings[2] trained on the Finnish Internet Parsebank (Kanerva et al., 2014). This word2vec model has been trained on lemmatized data, which is important as we are interested in obtaining replacement words in an uninflected form.

3.2 Genetic Algorithm

Genetic algorithms are inspired by evolution taking place in the real world. They have an initial set of individuals forming a population. These individuals are then exposed to evolutionary processes such as mutation and crossover. After a generation, the fittest individuals survive to the next generation and the evolutionary process is repeated. For modeling this process, we use the Python library DEAP (Fortin et al., 2012) as the genetic algorithms framework.

We employ a standard $(\mu + \lambda)$ genetic algorithm, which has previously been used in computational creativity applications (see Alnajjar et al.,

[1]https://fi.wikisource.org

[2]http://bionlp-www.utu.fi/fin-vector-space-models/fin-word2vec-lemma.bin

2018). The method begins by constructing an initial population and then evolving it, while optimizing certain parameters, throughout G generations. At each generation step, the fittest μ individuals in the current population and the λ offspring are selected to represent the next population. We empirically set μ and λ to 100 and G to 25. Additionally, the algorithm takes two user-defined inputs, a poem p and a theme t. For our case, we considered a theme t as a single word representing an abstract concept such as *nature*; alternatively, a set of words could be used instead to represent a more focused theme (e.g. *tree*, *forest*, *flower*, . . . etc).

3.2.1 Initial Population

To build an initial population containing poems with various syntactic structures, the method makes μ copies of the input poem p. For each poem, the method then replaces one verse in it with a random verse from a different poem existing in the poem corpus P.

3.2.2 Mutation and Crossover

In our method, we implement one type of mutation which selects a random content word in the poem. The term content word in this case refers to a word that belongs to an open class part-of-speech category. The selected word is substituted with another semantically similar word, which is determined as follows. Let w be the random content word selected to be replaced, the method retrieves the top 300 semantically similar words to w as candidate replacements from the word2vec model. Thereafter, the method uses UralicNLP (Hämäläinen, 2019) to perform morphological analysis on all candidate words. The candidate words that have a different part-of-speech tag than the original word w are omitted out. Out of the remaining candidate words, a random word is picked to substitute w.

We use a single-point crossover at the verse-level. In practice, this means that during the evolutionary process two poem individuals are selected and a single point at the beginning of their verses is chosen at random. Verses after that point are swapped between them.

3.2.3 Fitness Functions

The genetic algorithm assesses the individuals based on six metrics that evaluate the poem's structure and one metric that evaluates semantics. The difference in the number of syllables in verses and in the poetic foot, as measured by the distribution of long and short syllables, are contrasted to the original poem. The genetic algorithm is set to minimize these values to keep the difference minimal. As not changing the poem at all would result in the minimum difference in these values, we penalize identical verses by giving them a distance of 20. This way the genetic algorithm tries to make changes so that results following the original meter are preferred.

The number of full rhymes, assonance rhymes and consonance rhymes in between the verses of each generated poem are used as metrics to assess to overall poetic quality of the individuals. The number of alliterating words is counted within verses as this type of rhyme is traditionally occurring within verses in Finnish poetry. The values given by these four metrics are maximized by the genetic algorithm to get the maximum number rhyming words in the final outputs.

The last metric measures the average semantic similarity of the words in the poem to the input theme t with the word2vec model. Maximizing this function pushes the evolutionary process towards creating poems that are close in semantics to the desired input theme.

As we are employing multiple objective functions in our genetic algorithm, we resort to using a non-dominant sorting algorithm (NSGA-II) (Deb et al., 2002) for optimizing these functions. In short, the algorithm selects individuals that are not dominated by any other individual. An individual x is considered to be dominating another if its scores on all objective functions are greater than or equal to y's and it is always better than y on at least one objective.

3.3 Surface Generation

As the genetic algorithm does substitutions on the level of lemmas, it is important to be able to turn the verses with new lemmas into grammatical sentences. This is not only needed for presenting the final output produced by the genetic algorithm to people, but also for the fitness functions to work.

In Finnish, the surface form of a word (morphological realization) is affected by two mechanisms: agreement and government. The former means that certain words have to share morphological features in a sentence. For example, adjectives will have to follow the case and number of the noun they modify, like so: *punainen talo* (a red house) and *punaisessa talossa* (in a red house). This can be accounted for just by inflecting the replacement word with the morphology of the original word. For this purpose we use Omorfi (Pirinen et al., 2017) which implements Finnish morphology as an FST (finite-state transducer).

Government, on the other hand, requires some additional work. In government, words affect on each other morphologically in a way that depends on the governor. This means that if a governor word is replaced by another one in the sentence, the morphology of the governed word needs to adapt to the change. In concrete, given an original verse *uneksin hatusta* (I dream of a hat) and a change of the verb to *näen hatun* (I see a hat), the case of the object for *hattu* has to change from elative to genitive. We resolve government with Syntax maker (Hämäläinen and Rueter, 2018), which resolves the required case based on corpus statistics.

4 Results and Evaluation

As evaluation of creative systems is one of the most difficult problems in the field of computational creativity, instead of trying to come up with an evaluation metric of our own, we opt for the evaluation method used to evaluate a previous Finnish poem generator. In practice, this means conducting a quantitative evaluation with human judges with the evaluation questions defined by Toivanen et al. (2012).

An additional reasoning for using human evaluators instead of automated evaluation metrics is the poor correlation observed in a previous study (Hämäläinen and Alnajjar, 2019) of automatic evaluation metrics such as BLEU (Papineni et al., 2002) and PINC (Chen and Dolan, 2011) scores with human judgments when evaluating creativity of a system.

We run the genetic algorithm to produce a final population for 20 different initial poems for four different themes *luonto* (nature), *perhe* (family), *lemmikki* (pet) and *ihminen* (human). From each of the 20 final populations, we pick one poem at random. We shuffle the order of poems to reduce the priming effect of poems appearing always in a given order. We divide the 20 poems into two batches of 10 poems to reduce the effort of an individual evaluator. Each batch of 10 is then evaluated by 10 different human evaluators recruited from the university campus. The total number of evaluators is 20 and all of them are native in Finnish.

We use the following evaluation questions from Toivanen et al. (2012): (1) *How typical is the text as a poem?* (2) *How understandable is it?* (3) *How good is the language?* (4) *Does the text evoke mental images?* (5) *Does the text evoke emotions?* (6) *How much do you like the text?*. These questions are evaluated in a 5 point Likert scale, where 1 represents the worst and 5 the best grade. In addition to these questions, one simple binary question is asked: *Is the text a poem?*.

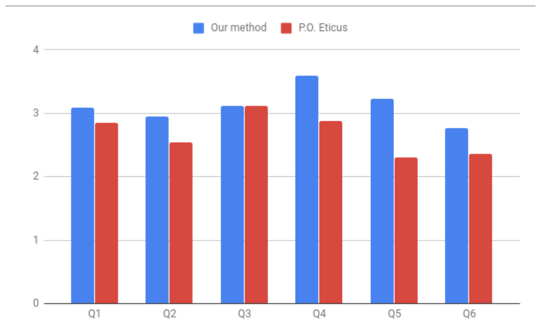

Figure 1: Evaluation results

Figure 1 represents the average values of the results of the human evaluation for each question. The plot also shows the evaluation results of P.O. Eticus as obtained in their study. As we can see, our method shows higher ratings on all the evaluation questions except for question 3. As for the binary question, the judges rated the output as a poem 81.5% of the time which is exactly the same result as P.O. Eticus got.

However, it is to remember that as a high level of subjectivity is involved in this evaluation setting, our results should not be directly compared to those of P.O. Eticus. The results form their study should taken more as a reference, rather than a definite proof that our system always outperforms P.O. Eticus.

	Q1	Q2	Q3	Q4	Q5	Q6
Average	3.10	2.94	3.11	3.60	3.23	2.77
Median	3	3	3	4	3	3
Mode	2	2	3	4	4	2

Table 1: The average, median and mode of the evaluation results

Table 1 shows the median and mode of the results in addition to the average values. The median values seems to correspond to the rounded average values. However, the mode deviates in the case of the first, second, fifth and last questions as the most chosen answer by the judges was different from the average.

Ja kultaa, kuninkaankin saan.
Ja laulut ne kiertävät maata ja merta
Jos virkkaan kun orja
Aina, todella Herra pahankurisuutta antaa.

And gold, of a king I shall have.
And songs, they shall roam on the land and the sea
If I knit like a slave
Always, indeed the Lord shall wrack his mischief.

Above is an example of a poem generated by the system and its translation in English. The poems generated by the system are typically of this length as the genetic algorithm uses a stanza of an existing poem as its starting point.

5 Discussion and Conclusion

The method presented in this paper shows improvement on a previously used evaluation metric. However, based on the discussions we had with some of the human evaluators after they had given their judgment, it became evident that people have very different criteria for poetry. Some of the judges had guessed that they were reading computer generated poems, even though this detail was not revealed to them explicitly. Their judgments were the most critical towards the generated poetry. On the other hand, the evaluators, who were struck by a surprise that the poems were generated by a computer, were in general more generous in their judgments. One of the evaluators almost refused to believe the poems were generated by a computer instead of a person.

The high level of subjectivity that we could observe just by talking with people calls for a more robust qualitative study on the poem evaluation problem itself in the future. This would allow us to uncover additional factors that affect on the judgments given by people. Furthermore, conducting a study just on the evaluation itself makes it possible for us to evaluate the adequacy of the used evaluation metric in evaluating computer generated poetry.

Nevertheless, the scores achieved by our system, in relation to a previous method by following the same evaluation metric, are promising as they are indicative of potentially higher quality in the output. We have presented a solution for the Finnish morphosyntax in conjunction with employing a genetic algorithm to cater for computational creativity in poem generation.

Acknowledgements

Special thanks to Jack Rueter for helping out with the evaluation.

References

Khalid Alnajjar, Hadaytullah Hadaytullah, and Hannu Toivonen. 2018. “Talent, Skill and Support.” A method for automatic creation of slogans. In *Proceedings of the 9th International Conference on Computational Creativity (ICCC 2018)*, pages 88–95, Salamanca, Spain. Association for Computational Creativity.

Benjamin Bay, Paul Bodily, and Dan Ventura. 2017. Text transformation via constraints and word embedding. In *Proceedings of the Eighth International Conference on Computational Creativity*, pages 49–56.

David L Chen and William B Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 190–200.

Simon Colton. 2008. Creativity Versus the Perception of Creativity in Computational Systems. In *AAAI Spring Symposium: Creative Intelligent Systems*, Technical Report SS-08-03, pages 14—-20, Stanford, California, USA.

Simon Colton, John William Charnley, and Alison Pease. 2011. Computational creativity theory: The face and idea descriptive models. In *ICCC*, pages 90–95.

Simon Colton, Jacob Goodwin, and Tony Veale. 2012. Full-face poetry generation. In *Proceedings of the Third International Conference on Computational Creativity*, pages 95—-102.

K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197.

Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner, Marc Parizeau, and Christian Gagné. 2012. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13:2171–2175.

Pablo Gervás. 2001. An expert system for the composition of formal Spanish poetry. *Knowledge-Based Systems*, 14(3):181–188.

Erica Greene, Tugba Bodrumlu, and Kevin Knight. 2010. Automatic analysis of rhythmic poetry with applications to generation and translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP ’10, pages 524–533, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mika Hämäläinen. 2018a. Harnessing NLG to Create Finnish Poetry Automatically. In *Proceedings of the Ninth International Conference on Computational Creativity*, pages 9–15.

Mika Hämäläinen. 2018b. Poem Machine - a Co-creative NLG Web Application for Poem Writing. In *The 11th International Conference on Natural Language Generation: Proceedings of the Conference*, pages 195—-196.

Mika Hämäläinen. 2019. UralicNLP: An NLP library for Uralic languages. *Journal of Open Source Software*, 4(37):1345.

Mika Hämäläinen and Khalid Alnajjar. 2019. Modelling the Socialization of Creative Agents in a Master-Apprentice Setting: The Case of Movie Title Puns. In *Proceedings of the Tenth International Conference on Computational Creativity*, pages 266–273.

Mika Hämäläinen and Jack Rueter. 2018. Development of an Open Source Natural Language Generation Tool for Finnish. In *Proceedings of the Fourth International Workshop on Computational Linguistics for Uralic Languages*, pages 51–58.

Katri Haverinen, Jenna Nyblom, Timo Viljanen, Veronika Laippala, Samuel Kohonen, Anna Missilä, Stina Ojala, Tapio Salakoski, and Filip Ginter. 2014. Building the essential resources for finnish: the turku dependency treebank. *Language Resources and Evaluation*, 48(3):493–531.

Raquel Hervás, Jason Robinson, and Pablo Gervás. 2007. Evolutionary assistance in alliteration and allelic drivel. In *Workshops on Applications of Evolutionary Computation*, pages 537–546. Springer.

Jenna Kanerva, Juhani Luotolahti, Veronika Laippala, and Filip Ginter. 2014. Syntactic n-gram collection from a large-scale corpus of internet Finnish. In *Human Language Technologies-The Baltic Perspective: Proceedings of the Sixth International Conference Baltic HLT*, volume 268, pages 184–191.

Anna Kantosalo, Jukka Toivanen, and Hannu Toivonen. 2015. Interaction Evaluation for Human-Computer Co-creativity: A Case Study. In *Proceedings of the Sixth International Conference on Computational Creativity*, pages 276–283.

Carolyn Lamb and Daniel G. Brown. 2019. TwitSong 3.0: towards semantic revisions in computational poetry. In *Proceedings of the Tenth International Conference on Computational Creativity*, pages 212–219.

Juntao Li, Yan Song, Haisong Zhang, Dongmin Chen, Shuming Shi, Dongyan Zhao, and Rui Yan. 2018. Generating classical Chinese poems via conditional variational autoencoder and adversarial training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3890–3900, Brussels, Belgium. Association for Computational Linguistics.

Ruli Manurung, Graeme Ritchie, and Henry Thompson. 2012. Using genetic algorithms to create meaningful poetic text. *J. Exp. Theor. Artif. Intell.*, 24:43–64.

Hugo Gonçalo Oliveira. 2017. A survey on intelligent poetry generation: Languages, features, techniques, reutilisation and evaluation. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 11–20, Santiago de Compostela, Spain. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318.

Tommi A Pirinen, Inari Listenmaa, Ryan Johnson, Francis M. Tyers, and Juha Kuokkala. 2017. Open morphology of finnish. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University.

Jukka Toivanen, Hannu Toivonen, Alessandro Valitutti, and Oskar Gross. 2012. Corpus-Based Generation of Content and Form in Poetry. In *Proceedings of the Third International Conference on Computational Creativity*.

Geraint A Wiggins. 2006. A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-Based Systems*, 19(7):449–458.