Automatic Classification of Poetry by Neural Scansion

Hyungrok Kim, Yu Zhao, and Belay Zelalem
School of Computing, KAIST
Daejeon, Republic of Korea
{q0115643, zhaoyu, zelalem.mihret}@kaist.ac.kr

***Abstract*—Recent researches has been focusing on large scale poetry classification by meter. We have found significant problems in poetry that can be hard to be resolved by rule-based poetry scansion programs. In this paper, we tackle this problem by replacing formal poetry scansion programs by neural network approach, the ”Neural Scansion.” By our machine learning approach, we built our model to syllabify the words in poems. Per syllable, we generated corresponding, predicted poetic meter, and made classifications within the generated poetic meters. With these experiments, our purpose is establish a new baseline of automatic poetry neural scansion handling unexpected deviations from such standard patterns.**

I. Problem

As we are in the digital age, huge amount of available data has attracted the attention of major scholars and has developed into its own research paradigm. There is no consensus as to when data are large or complex enough to qualify as the object of data-intensive research, especially since huge or massive may mean completely different things in different fields and disciplines, but Levallois, Steinmetz, and Wouters advance a relevant and potentially useful definition: data-intensive research [is] research that requires radical changes in the discipline involving “new, possibly more standardized and technology-intensive ways to store, annotate, and share data,” a concept that therefore “may point toward quite different research practices and computational tools” [1].

[2] is a project proposal that was focusing on studying to redefine the scholarly approach to poetry analysis by applying data-intensive research methods and eventually mathematical graph theory. Following researches ([3], [4], [5], [6]) study the features of graphs and find some patterns, conclusions against the established literary criticism.

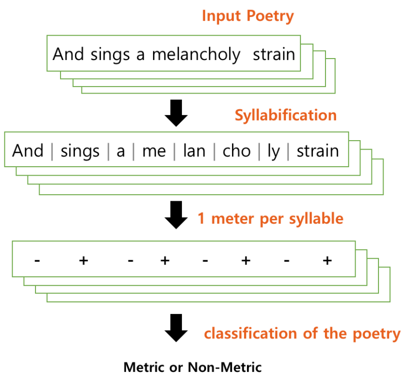

Input Poetry
And sings a melancholy strain
Syllabification
And | sings | a | me | lan | cho | ly | strain
1 meter per syllable
- + - + - + - +
classification of the poetry
Metric or Non-Metric

Fig. 1. Progress of poetry classification by meters

In a recent research [3], the authors worked on classifying English poems into metric or non-metric, by scanning and analyzing their phonetic patterns with poetic meters and rhymes, like in 1. Poem is about poetic meter, line structure, and use of rhyme. In [3], the authors extracted metrical related features from raw text, which are the poetic meters. The poetic meters are stressed or unstressed accent on each syllable, so to extract the meters from a raw poetry, we need to split the words into syllables, which is called ‘syllabification.’

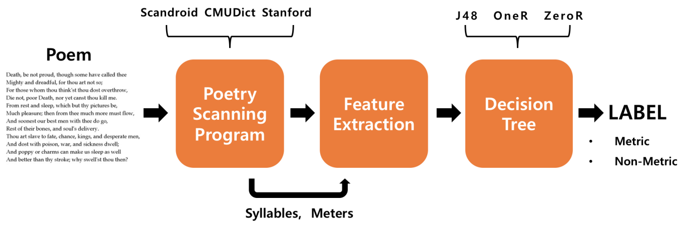

Scandroid CMUDict Stanford
J48 OneR ZeroR
Poem
Death, be not proud, though some have called thee
Mighty and dreadful, for thou art not so;
For those whom thou think'st thou dost overthrow,
Die not, poor Death, nor yet canst thou kill me.
From rest and sleep, which but thy pictures be,
Much pleasure; then from thee much more must flow,
And soonest our best men with thee do go,
Rest of their bones, and soul's delivery.
Thou art slave to fate, chance, kings, and desperate men,
And dost with poison, war, and sickness dwell;
And poppy or charms can make us sleep as well
And better than thy stroke; why swell'st thou then?
Poetry Scanning Program
Feature Extraction
Decision Tree
LABEL
Metric
Non-Metric
Syllables, Meters

Fig. 2. Detailed techniques used in [3]

2 shows the specific progress in [3], with automatic rule-based poetry scansion program “Scandroid” [7], they extracted syllables and poetic meters from raw poetry, and after feature extraction process, the decision tree decides whether the input poem is metric or non-metric.

From the scansion process of Scandroid [7], we spotted two significant problems that need to be solved. First in the syllabification, problems come with poetic Licenses and unknown words. In poems, awkward words based on poetic license exist frequently, Rule-based algorithm cannot work as a perfect syllabification. Second in meter generation, lexical ambiguity is a significant problem. Some words meters differ depending on whether they are verbs or nouns, for example word “convict” has different accent structures like conVICT(verb) or CONvict(noun).

The two problems are not easy to be solved with rule-based algorithm that scansion programs like [7] have, and those problem affects the quality of generated meters. So we came up with new approach by neural network approach to provide better meter extraction. For syllabification and meter extraction, we will compare our new approach and the baseline method from [3] and eventually the effects in classification phase.

II. Related work

Computational analyses of small, manually labeled corpora of poetry have been conducted before, notably by Malcolm

Hayward [8]. In 1991, Hayward published a paper on a connectionist constraint satisfaction model based on a parallel distributed processing model for analyzing the metrical features of poetry. He was motivated by traditional metrical analytic methods inability to account for interactions between all aspects of a poem and their overall effect on prosody during reading performances.

Hayward divided the integral of activation in the soft position by the integral of activation in the hard position to measure metrical regularity in iambic verse. Ratios close to 0 indicated a more regular iambic verse while ratios that approached 1 indicated a less regular use of iambic pentameter (or none at all) [8].

Convinced that this model could be used to uncover individual poets unique metrical style, in 1996 Hayward published a follow up paper comparing 10 poets treatment of iambic verse and was able to fingerprint them and determine their period of writing [9].

While sophisticated, Haywards approach can not scale as it relies on hand annotated data. Additionally, Hayward analyzed only 1000 lines of poetry and it is unknown how well his model will generalize for different corpora.

More recently, researchers in the digital humanities have begun working with much larger datasets. [10] from the Stanford Literary Lab performed data intensive poetry analysis via the Trans-Historical Literary Project. As of this writing they have not published a paper, however an abstract of their work is available from dharchive.org [10] and their source code is available on Heusers github page [11]. According to their presentation notes (available on github), their parser was able to correctly divide words into syllables with an accuracy of 88%. Still, because the approach by [10] was based on rule-based algorithms, it should be compared in our results.

In 2017, from [12], there was an neural net approach to phonetic model of poetry, but it was for generating verses of poetry, not the classification as our baseline approach from [3], but since the meter extraction part of our process is like meter generation, which is generation of phonetic sequence like in [12], the model they used, should be in trial and compared.

III. Solution

Basic difference of the new approach is that we are applying the neural network approach instead of the rule-based poetry scansion program, so we are building and comparing the Neural Scansion with the scansion program from baseline [3].

There are two tasks that the neural scansion should solve, first is the syllabification from raw poetry input, and the second in meter extraction from the syllables that was the output of the first phase (syllabification). We must extract syllables before generating meters but explanation of meter generation part will be first for convenience.

A. Meter Generation

To apply neural net model to meter generation, first we thought of this as a sequence-to-sequence problem sequence of syllables to sequence of meters. So we can model it just like the rhythmic verse generating paper we mentioned [12]. We tried building the encoder-decoder LSTM [13] approaches [14] but it didnt give us any good performance, especially the length of sequence were not matching. We thought of it as a tagging method, because each syllable becomes one meter, we thought of it as tagging a meter per syllable. And used the state of the art model in named entity recognition, Bi-LSTM-CRF model [15].

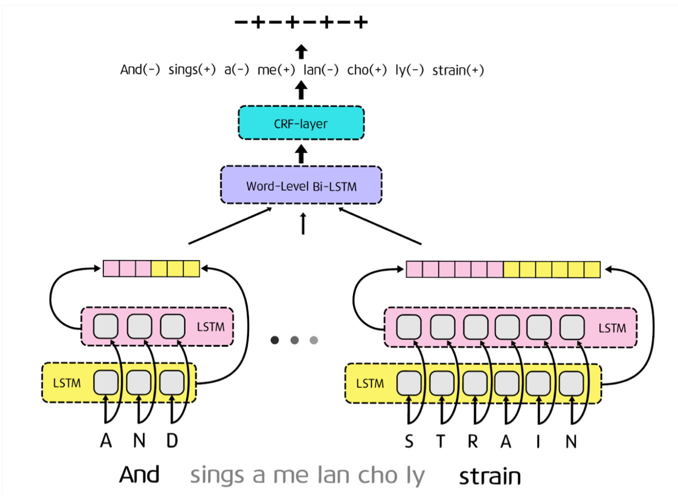


Fig. 3. Bi-LSTM-CRF model for meter generation

3 is a closer look to our Bi-LSTM-CRF model for meter generation. Each syllable is encoded into character-level bi-LSTM and generates syllable vector by concatenating two outputs. After, sequence of those syllable vectors become inputs for word-level Bi-LSTM, and into CRF layer, we can get sequence of corresponding meters.

The original model from [15] uses pre-trained word vectors to be concatenated with the character-level LSTMs' outputs, but since we are using syllables as tokens, there aren't any available pre-trained syllable vectors we can use, so we removed that part.

With this model, there are some variations we could make. We could include the special characters like question mark and dot, which can hold information about the meter it has. And because we use syllable-level model, we lose the original word structures so we can added word boundary characters to the syllables.

B. Syllabification

4 is a closer look to our Bi-LSTM-CRF model for syllabification. Basically we used the same model as 3. Important thing was that how we made the syllabification into a tagging task, since this model 3 was for tagging task. We tokenized each word into characters-level, and tagged 'b' or 'i', 'b' meaning beginning of a syllable and 'i' for inside a syllable. This way, we made syllabification into a tagging task.

If you look at 4, the character-level LSTM might not be useful because the input is just one token of a character, actually the performance doesnt differ much whether using character-level LSTM or not.

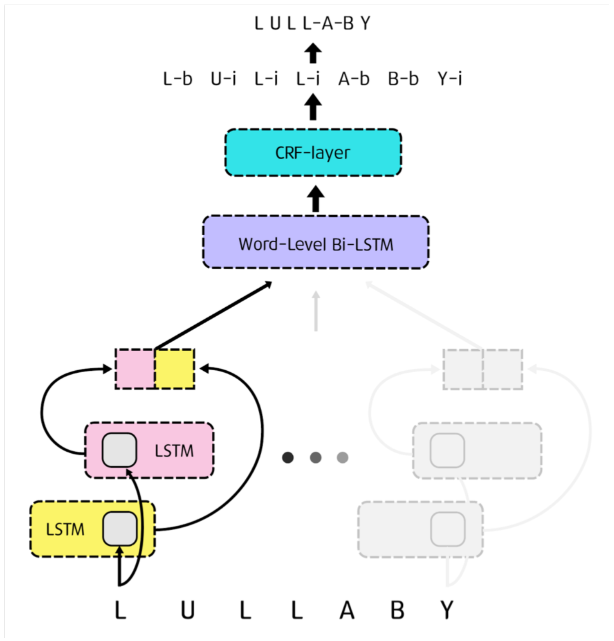
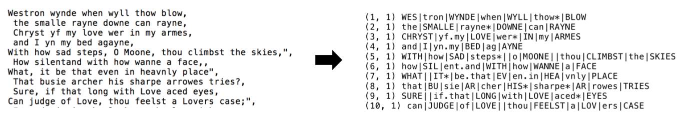


Fig. 4. Bi-LSTM-CRF model for syllabification

IV. Experiments

For our experimental setup, we implemented our neural scansion and the baseline model [3] from scratch since the authors didn't give enough setups or any of their code. First for the baseline model, we had to prepare the dataset, scansion program called Scandroid [7], and Feature Extraction process for the decision tree and the decision tree.

For dataset, we crawled 5840 categorized poems from poetryfoundation website[1]. For scansion program, we found that Scandroid [7] was not really handy to use, so we found another similar program called LitLab-poetry [10] from Stanford. like in 5, with input poem, this program syllabify words and marks the stressed and unstressed meters, the capital syllables mean stressed and others are not.



```
Westron wynde when wyll thow blow,
 the smalle rayne downe can rayne,
 Chryst yf my love wer in my armes,
 and I yn my bed agayne,
With how sad steps, O Moone, thou climbst the skies,",
 How silentand with how wanne a face,,
What, it be that even in heavnly place",
 That busie archer his sharpe arrowes tries?,
 Sure, if that long with Love aced eyes,
Can judge of Love, thou feelst a Lovers case;",
```

```
(1, 1) WES|tron|WYNDE|when|WYLL|thow*|BLOW
(2, 1) the|SMALLE|rayne*|DOWNE|can|RAYNE
(3, 1) CHRYST|yf.my|LOVE|wer*|IN|my|ARMES
(4, 1) and|I|yn.my|BED|ag|AYNE
(5, 1) WITH|how|SAD|steps*||o|MOONE||thou|CLIMBST|the|SKIES
(6, 1) how|SIL|ent.and|WITH|how|WANNE|a|FACE
(7, 1) WHAT||IT*|be.that|EV|en.in|HEA|vnly|PLACE
(8, 1) that|BU|sie|AR|cher|HIS*|sharpe*|AR|rowes|TRIES
(9, 1) SURE||if.that|LONG|with|LOVE|aced*|EYES
(10, 1) can|JUDGE|of|LOVE||thou|FEELST|a|LOV|ers|CASE
```

Fig. 5. working example from LitLab-poetry program [10]

For the feature extraction and decision tree they [3] used, since the authors didnt provide the implementation of those nor specific details about it, we replaced that part with neural network classifier which can classify poems as metric or non-metric with input data of meters. We chose do this because we are focusing on the effect of the syllable and meter extraction model, the classifier didnt matter because we used same classifier for both approaches.

[1]http://poetryfoundation.org

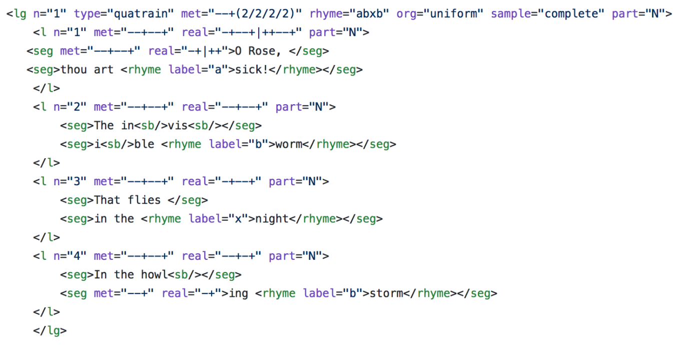
And for our neural network approach, We needed to implement two parts, first the syllable extraction, and second the meter generation.

A. Syllabification

First for syllabification, as we said, We used Character-level Bi-LSTM-CRF model 4, and for training, we collected some syllable dictionary from web database called WebCelex [16], which has 160,595 words with full syllable information. We found out the poems from poetryfoundations [17] data have about 34,000 words included in the dictionary and 25,000 as unknown words. So we trained the model with trainset including known words and 3,000 for devset and testset.

The dimensions for character-level LSTMs were set to 27, and 128 for word-level LSTMs and both were accompanying one another LSTM to form bidirectional LSTMs. Stochastic gradient Descent [18] optimizer was used with learning rate of 0.005 and dropout layer [19] with dropout rate of 0.5.

B. Meter Generation



```
<lg n="1" type="quatrain" met="--+(2/2/2/2)" rhyme="abxb" org="uniform" sample="complete" part="N">
   <l n="1" met="--+--+" real="-+--+|++--+" part="N">
  <seg met="--+--+" real="-+|++">O Rose, </seg>
  <seg>thou art <rhyme label="a">sick!</rhyme></seg>
   </l>
   <l n="2" met="--+--+" real="--+--+" part="N">
       <seg>The in<sb/>vis<sb/></seg>
       <seg>i<sb/>ble <rhyme label="b">worm</rhyme></seg>
   </l>
   <l n="3" met="--+--+" real="-+--+" part="N">
       <seg>That flies </seg>
       <seg>in the <rhyme label="x">night</rhyme></seg>
   </l>
   <l n="4" met="--+--+" real="--+-+" part="N">
       <seg>In the howl<sb/></seg>
       <seg met="--+" real="-+">ing <rhyme label="b">storm</rhyme></seg>
   </l>
   </lg>
```

Fig. 6. example paragraph from 4B4V dataset [20]

For Meter generation, we used the same model 3, and here because we needed gold labels syllables and meters in training, we found this 4B4V dataset [20] with 87 poems, in total of about 12 hundred lines with all features like in 6, it has all syllables and meters.

All setup was similar to syllabification, but the dimensions in character-level bi-LSTMs were set to 31.

C. Metric Classifier

With those 2 phase in our approach, we made a neural replacement against the rule-based scansion program. After, we used CNN and RNN classifier for classifying the poems into metric or non-metric. CNN was used with the model introduced by Yoon Kim in [21], and RNN was just used with one simple layer structure of GRU [22].

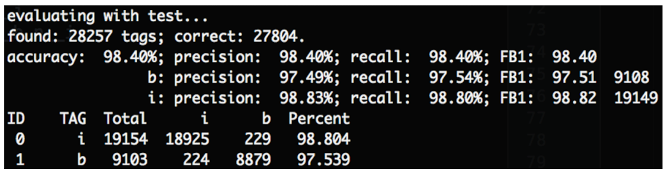
For convolutional model from [21], the kernel sizes were set to {2, 3, 4, 5}, dropout rate was set to 0.5, and due to computational overhead, we only used static word embedding with learning rate 0.001 and batch size 128.

For GRU model from [22], the hidden dimension was set to 100, batch size 64 and we used Adam optimizer from [23] with learning rate of 0.001.

V. Results

Within the results, we made 3 comparisons. First, the syllabification accuracy per word comparing our neural network model with the Litlab-poetry scansion [10]. Second, meter generation result on 4B4V data [20], we compared the syllable-level accuracy, line-level accuracy, and the accuracy of the line lengths. For final result for poetry classification on poetryfoundation data [17], we only compared the whole output accuracy as metric or non-metric.

A. Syllabification



```
evaluating with test...
found: 28257 tags; correct: 27804.
accuracy:  98.40%; precision:  98.40%; recall:  98.40%; FB1:  98.40
               b: precision:  97.49%; recall:  97.54%; FB1:  97.51  9108
               i: precision:  98.83%; recall:  98.80%; FB1:  98.82  19149
ID    TAG  Total      i      b  Percent
 0      i  19154  18925    229   98.804
 1      b   9103    224   8879   97.539
```

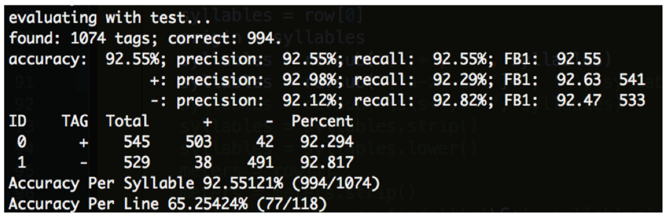
Fig. 7. syllabification accuracy per character-tag (b, i)

Accuracy	per character-tag	per word-to-syllables
LitLab-poetry	-	88.00%
Bi-LSTM-CRF	98.40%	91.90%

TABLE I
COMPARISONS OF ACCURACY RESULTS OF SYLLABIFICATION

7 and I show the accuracy results and comparisons with baseline program and our neural scansion. The LitLab-poetry's syllabification accuracy was given in their presentation notes (available on github). We have drawn higher accuracy than the baseline rule-based scansion, which became 91.90% from 88.00%.

B. Meter Generation



```
evaluating with test...
found: 1074 tags; correct: 994.
accuracy:  92.55%; precision:  92.55%; recall:  92.55%; FB1:  92.55
               +: precision:  92.98%; recall:  92.29%; FB1:  92.63  541
               -: precision:  92.12%; recall:  92.82%; FB1:  92.47  533
ID    TAG  Total      +      -  Percent
 0      +    545    503     42   92.294
 1      -    529     38    491   92.817
Accuracy Per Syllable 92.55121% (994/1074)
Accuracy Per Line 65.25424% (77/118)
```

Fig. 8. meter extraction accuracy per syllable and per line

Accuracy	per syllable	per line	line length
LitLab-poetry	72.00%	27.01%	81%
Bi-LSTM-CRF	92.08%	62.71%	100%
Bi-LSTM-CRF with (?, !, .)	92.15%	63.80%	100%
Bi-LSTM-CRF with (?, !, .), WB	92.55%	65.25%	100%

TABLE II
COMPARISONS OF ACCURACY RESULTS OF METER GENERATION

8 and II(WB: word-boundaries to bring back the word structural information) show the accuracy results and comparisons with baseline program and our neural scansion. Here also, we can see that our neural scansion approach has got much higher accuracy than the rule-based scansion program.

C. Final Result for Poetry Classification

Accuracy for	CNN	RNN
Our Neural Scansion	72%	76%
Scansion Program (LitLab-poetry)	75%	78%

TABLE III
COMPARISONS OF ACCURACY RESULTS OF SYLLABIFICATION

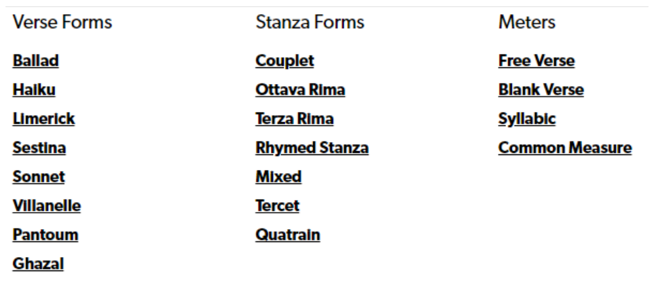
Odd results compared to two previous results, III shows that our neural scansion approach got slightly lower accuracy than the baseline model for poem classification to metric or non-metric.

VI. Discussion

A. Conclusion

Neural Scansion for syllabification was better than baseline program's rule-based algorithm or dictionary, since it's probabilistic, machine learning approach had great ability to learn the changes of pronunciations of syllables.

Even for Meter Generation, our neural scansion had much better results with higher accuracy. Especially for accuracy per line, it had improved score from 27.09% to 65.25%. Also for line length, 81% of results with LitLab-poetry had wrong length of output meter sequence, but because we adapted tagging model to our task, we were able to have 100% accurate meter sequence lengths. And the accuracy got increased as it included special characters like ?, !, ., and word boundaries. This should be because the special characters like dot, question mark, and exclamation mark have information of accents, which is directly connected to the meter structure, and for the word boundaries, it will bring back the word form into our neural network because by using syllable-level model, we might lose the word structures.



Verse Forms	Stanza Forms	Meters
Ballad	Couplet	Free Verse
Haiku	Ottava Rima	Blank Verse
Limerick	Terza Rima	Syllabic
Sestina	Rhymed Stanza	Common Measure
Sonnet	Mixed	
Villanelle	Tercet	
Pantoum	Quatrain	
Ghazal		

Fig. 9. structural categories of poems from poetryfoundation.org

The final poetry classification result was similar to the baseline model and was not higher, this was unexpected because the previous two experiments had significantly higher performance, this might be caused by the following reasons,

- The difference between 4B4V and Poetry dataset, because the 4B4V dataset [20] was very small, only with 87 poems, it would have been hard to train the model to have metric or non-metric features correctly.

- Better capacity of Decision Tree than CNN/RNN on classifier, since the structural categories are directly related to meters and rhyme structures, there will not be many exceptional cases that could had been handled better by neural networks, so it would have been messing the results because of using too simple neural network approach in the final metric or non-metric classification task (we used basic CNN [21] or RNN [22] method).
- Ambiguous standard of Metric or Non-Metric, we expect this reason as the most significant one. The poetry categorizing is difficult to just say Metric or Non-Metric if we look at 9, some categories like "Free Verse" should be non-metric, but there are no simple, general rule to classify metric and non-metric poems since the categories of the poetry became diverse with the postmodern literature. The baseline paper [3] doesn't have detailed explanations how they picked which one as metric and which one as non-metric when they were building their dataset.

B. Future Work

In this paper, we worked on neural network approach of meter generation and classification of poetry based on poetic meters. But the actual poetry structural categories are based on meters and rhymes, so with additional Rhyme Detection and Poetry structure understandings for all categories, we will be able to get improve and very precise the metrical analysis not just by (metric, non-metric), classifying all categories like in 9.

In our model, we divided the syllabification task and meter generation task in separate progress, but if we jointly model the poetic language, meter and rhyme, we might have compact and effective neural model for analyzing poems.

References

[1] C. Levallois, S. Steinmetz, P. Wouters *et al.*, "Sloppy data floods or precise social science methodologies? dilemmas in the transition to data-intensive research in sociology and economics," *Virtual Knowledge: Experimenting in the Humanities and the Social Sciences*, pp. 151–182, 2013.

[2] MARGENTO, "The graph poem project," 2015, http://artsites.uottawa.ca/margento/en/the-graph-poem/.

[3] C. Tanasescu, B. Paget, and D. Inkpen, "Automatic classification of poetry by meter and rhyme." in *FLAIRS Conference*, 2016, pp. 244–249.

[4] M. Pramanick, A. Gupta, and P. Mitra, "An lstm-crf based approach to token-level metaphor detection," in *Proceedings of the Workshop on Figurative Language Processing*, 2018, pp. 67–75.

[5] A. Lou, D. Inkpen, and C. Tanasescu, "Multilabel subject-based classification of poetry." in *FLAIRS Conference*, 2015, pp. 187–192.

[6] V. Kesarwani, D. Inkpen, S. Szpakowicz, and C. Tanasescu, "Metaphor detection in a poetry corpus," in *Proceedings of the Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, 2017, pp. 1–9.

[7] Hartman, C, "Charles hartman programs," 2004, http://oak.conncoll.edu/cohar/Programs.htm/.

[8] M. Hayward, "A connectionist model of poetic meter," *Poetics*, vol. 20, no. 4, pp. 303–317, 1991.

[9] ——, "Applications of a connectionist model of poetic meter to problems in generative metrics," in *Research in Humanities Computing: Selected Papers from the ALLC/ACH Conference*, vol. 4, 1996, pp. 185–92.

[10] M. Algee-Hewitt, R. Heuser, M. Kraxenberger, J. Porter, J. Sensenbaugh, and J. Tackett, "The stanford literary lab transhistorical poetry project phase ii: Metrical form," in *Digital Humanities Conference, Lausanne, Switzerland*, 2014.

[11] Heuser, R, "Stanford literary lab github account," 2015, https://github.com/quadrismegistus/litlab-poetry.

[12] J. Hopkins and D. Kiela, "Automatically generating rhythmic verse with neural networks," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2017, pp. 168–178.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[15] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," *arXiv preprint arXiv:1603.01360*, 2016.

[16] R. Baayen, R. Piepenbrock, and L. Gulikers, "Webcelex," 2001.

[17] K. Goldsmith, "Flarf is dionysus. conceptual writing is apollo. an introduction to the 21st century's most controversial poetry movements." poetryfoundation. org," *Poetry Foundation*, vol. 1, 2009.

[18] H. Robbins and S. Monro, "A stochastic approximation method," *The annals of mathematical statistics*, pp. 400–407, 1951.

[19] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[20] H. F. Tucker, "Poetic data and the news from poems: A for better for verse memoir," *Victorian Poetry*, vol. 49, no. 2, pp. 267–281, 2011.

[21] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[22] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.