Predicting Concrete and Abstract Entities in Modern Poetry

Fiammetta Caccavale, Anders Søgaard
University of Copenhagen
Universitetsparken 1
DK-2100 Copenhagen

Abstract

One dimension of modernist poetry is introducing entities in surprising contexts, such as *wheelbarrow* in Bob Dylan's *feel like falling in love with the first woman I meet/ putting her in a wheelbarrow*. This paper considers the problem of teaching a neural language model to select *poetic entities*, based on local context windows. We do so by fine-tuning and evaluating language models on the poetry of American modernists, both on seen and unseen poets, and across a range of experimental designs. We also compare the performance of our poetic language model to human, professional poets. Our main finding is that, perhaps surprisingly, modernist poetry differs most from ordinary language when entities are *concrete*, like *wheelbarrow*, and while our fine-tuning strategy successfully adapts to poetic language in general, outperforming professional poets, the biggest error reduction is observed with concrete entities.

Introduction

Paul Valéry defines the "language of poetry" as a "language within a language" (Valéry 1957), alluding to the fact that poetic language does not necessary reflect the conventions of every-day language. For the same reason, it is relatively easy for most people to detect poetic language, even in the absence of rhyme and unconventional word order. At the same time, poetic language is considered mysterious and impenetrable by many, and often also idiosyncratic.

Poets often spend most of their energy on *labor limae*, deciding which words are more poetical, expressive and successful in conveying the intended idea. Poets are able to find the most *poetically interesting* word for a given context based on their subtle and nuanced grasp of the slight variations in meaning that each word bears upon the text.

Several poets have tried to define what they mean by 'poetically interesting'; and to mention just one example, the American poet Charles Bernstein has described the poetically interesting as *anti-absorptive* (Bernstein 1987). Anti-absorption means that a poetically interesting word or entity suggests ways of interacting with its context, but does not simply blend in. As a reader you stumble – the entity feels misplaced, somehow – but at the same time, the entity and the context align along *some* dimensions, suggesting several, possible readings. In effect, form becomes visible - no longer a simple means to convey content. This is in sharp contrast with prose or scientific writing, of course, in which form is always a slave of content.

We believe entities (or common nouns), and how they are introduced, constitute a key dimension of modern poetry. Part of this is a willingness of modern poets to introduce entities in unusual contexts (*odd pairings*); part of it is the use of entities in novel metaphor schemes. The word *sea*, for example, usually refers to a vast extension of water, but in Van Jordan's poem "Vestiges", *sea* is a metaphor of time, introducing characteristics of coldness and depth, as well as the notion of voyage.

> A wind breathes across the sea,
> joining gently the edges of time.

This paper asks whether a neural network is capable of learning such poetic decisions in ways that generalize across different poets? Specifically, we investigate whether neural language models can learn to predict entities in modern poetry based on the preceding context, and successfully do so on texts by previously unseen poets.

Specifically, we consider the task of reconstructing the original verse of a poem, extracted from a corpus of American modern poetry, and evaluate models for this task across different poets. Our baseline model is a state-of-the-art neural language model, trained on ordinary language (Wikipedia). Note that it is an open question whether modeling poetic choices is *at all learnable* or not. If poetic choices are entirely idiosyncratic (Java 2015), predicting the choices of unseen authors beyond chance should be impossible.

Predicting the exact word used by a poet out of a vocabulary of hundreds of thousands of words is extremely hard, but we consider the task of selecting the right candidate from a set of related words, e.g., a set of synonyms or words of the same high-level semantic class. We evaluate our models on held-out sentences, possibly from held-out poets. Finally, we compare the performance of our models to human, professional poets. [1]

[1] Note the idea of evaluating poetry, in part, by how much it departs from our internal language models (conven-

Background

NLP for poetry NLP for poetry has mainly focused on stylistic analysis (Hayward 1996; Kaplan and Blei 2007; He et al. 2007; Fang, Lo, and Chinn 2009; Greene, Bodrumlu, and Knight 2010; Genzel, Uszkoreit, and Och 2010; Kao and Jurafsky 2012) and poetry generation (Manurung, Ritchie, and Thompson 2012; Zhang and Lapata 2014; Ghazvininejad et al. 2016). Research on stylistics has focused on the features that make a poem come across as poetic (Kao and Jurafsky 2012); on quantifying poetic devices such as rhyme and meter (Hayward 1996; Greene, Bodrumlu, and Knight 2010; Genzel, Uszkoreit, and Och 2010); on evidence of intertextuality and how to prove stylistic influence between authors (Forstall, Jacobson, and Scheirer 2011); or on authorship and style attribution (Kaplan and Blei 2007; He et al. 2007; Fang, Lo, and Chinn 2009). These studies are examples of detecting statistical regularities in poetic language, potentially helping us to better understand and categorize poetic literature (Fabb 2006). Note, however, that the vast majority of previous work has focused on *classical* poetry, where poetic markers are often associated with rhyme and meter. We instead focus on modern poetry, where, in contrast, such markers are typically absent.

Our work is perhaps most related to Kao and Jurafsky (2012), but we focus on a slightly different, more narrow problem, namely that of modeling how entities are introduced in modern poetry. We approach this problem as a language model fine-tuning task.

Fine-tuning neural language models Fine-tuning neural language models is a simple form of transfer learning, in which training relies on pretrained language model weights and early stopping to adapt the pretrained weights to a new task or domain. In Bayesian terms, we induce a neural language model for poetry using a common language model as a *prior*. Fine-tuning has been used to adapt neural language models to new domains in neural machine translation for years (Luong, Kayser, and Manning 2015). In neural machine translation, similar effects can also be obtained by directly training sequence-to-sequence architectures on monolingual data in an auto-encoding fashion (Sennrich, Haddow, and Birch 2016).

Some authors have argued that fine-tuning is only robust when using an elaborate scheme of layer-wise unfreezing (Felbo et al. 2017; Howard and Ruder 2018), but in this paper we show that a simple, straight-forward application of fine-tuning can be used to adapt a language model trained on Wikipedia to poetic language. Similar applications of fine-tuning have previously been used to adapt language models to sign language (Mocialov, Turner, and Hastie 2018), for example.

Evaluating language models when ranking vocabulary subsets We will not be interested in the general perplexity of our language models, but rather the ability of our models to choose words referring to poetic entities from a set of comparable or similar entities. This is motivated by the assumption that poets typically choose entities from a small set of semantically consistent words. Constraining the space of possible outputs of a language model is not a novel idea. Mi, Wang, and Ittycheriah (2016) use this technique to reduce computing time and memory usage in neural machine translation, by introducing a sentence-level or batch-level vocabulary, which is only a small subset of the full output vocabulary. The method selects a small vocabulary of possible target words for each sentence, based on a word-to-word translation model or a bilingual phrase library, learned from a traditional machine translation model. In L'Hostis, Grangier, and Auli (2016), the output vocabulary of a neural translation model is restricted to a subset of likely candidates given the source.

tionality) has been floated by many literary scholars, see, e.g., https://www.theguardian.com/books/booksblog/2014/feb/20/perfect-word-poem-oxford-practical-criticism.

Data

Main data Our dataset was collected using the public API PoetryDB (PoetryDB). The total number of poets that can be retrieved through the API is 129, but given that we restrict ourselves to American poetry, only the poems of American poets were collected, leaving us with a dataset of 32 authors and 750 poems. After cleaning the data and removing sentence duplicates, the corpus results in 33,118 sentences. This is a relatively small dataset, making domain adaptation a more challenging task.

Pre-processing The data is tokenized, reduced to lower case, and punctuation is removed from the sentences. We identify all nouns in the sentences, excluding pronouns and names, by assigning a part of speech tag to each token in the corpus. The nouns will be the targets we try to predict, and the context windows will consist of words occurring on both sides, as limited by verse boundaries.

Pretrained word embeddings We use pre-trained word embeddings to vectorize words, associating each word with a vector representation reflecting its distributional properties in large corpora of ordinary language (typically Wikipedia). General-domain pre-trained word embeddings biases our model toward ordinary language (Kesarwani 2018), but also many words in the poetry domain may not occur in Wikipedia. Not to put our baseline model at a disadvantage, we train our word embeddings on the concatenation of our Wikipedia and training data of our poetry dataset. We train our word embeddings using word2vec with hyperparameters dimensionality of 100, window size of 5, and no frequency cut-off.

Annotations by professional poets For 75 held-out examples, we constructed a multiple choice questionnaire that we asked four professional poets to fill out. The poets received the following instructions: *You will be given 75 verses extracted from American poems. Each verse presents a missing noun: the task is to select the noun that, in your opinion, fits better.* The inter-annotator agreement score among the

poets was calculated using a measure called Fleiss' Kappa. The Kappa value in the annotations is: 0.332, which is quite low. We use this data below in Experiment III, where we show that our fine-tuned neural language model is better at predicting the concrete and abstract entities used by the original authors than the poets. See the relevant section for discussion.

Methodology

Our experiments are designed to compare the performance of a neural language model fine-tuned on sentences from poems written by modern American poets to a state-of-the-art neural language model trained on ordinary language (our baseline), on the task of predicting concrete and abstract entities in modern poetry.

Baseline and pretraining

The baseline is provided by a bidirectional LSTM (Hochreiter and Schmidhuber 1997), trained on Wikipedia. We use a publicly available implementation for our baseline,[2] and modify the same code for our system, which we make publicly available here.[3] The Wikipedia corpus dump used in this study is publicly available.[4] The resulting model contains 17.3M parameters. This pre-trained model will be used to transfer knowledge to the second model, which is referred to in this study as the *fine-tuned model*. Both models are bidirectional LSTMs.

All hyper-parameters of our models were tuned to minimize perplexity on heldout Wikipedia data (putting our baseline at an advantage). Our architecture is the following: the first layer is an embedding layer, the second is a bidirectional RNN with an LSTM cell. The third is a softmax layer. The embedding layer has input dimension equal to the vocabulary size (55395), the output dimension specified is the embedding size, which is the length of each embedding layer (100), the input length is set to 55, which is the length of the longest sentence in the corpus (all the other sentences have been padded to the same length), and, lastly, the embedding weights previously saved are loaded, so the network does not have to randomly initialize the weights. The bidirectional LSTM layer has 200 neurons. The units in the dense layer are equal to the vocabulary size, since the aim is to return a probability for each word in the corpus.

Training details: The back-propagation algorithm, which optimizes the network parameters using root mean square propagation (RMSprop), is run for 10 iterations.The training batch size is set to 32, and the pre-training learning rate used is 0.001.

Fine-tuning

Transfer learning refers to the idea that knowledge induced from training on one task or one source of data can be adapted or transferred to a model for another task or source of data. Fine-tuning is perhaps the simplest form of transfer learning (Pan, Yang, and others 2010).

Typically, fine-tuning refers to the idea of training a model – in our case, a neural network – to learn parameters, say representations, for a broad domain or a task where data is abundant, minimizing error in that domain; and then subsequently, optimizing the model to minimize the error in another domain, often more specific than the one used in the previous stage. Very often the second round of training, i.e., the fine-tuning, uses some regularization technique or heuristic to avoid catastrophic forgetting (McCloskey and Cohen 1989), e.g., early stopping (Girosi, Jones, and Poggio 1995). Fine-tuning thereby allows for features or model parameters to be transferred from the broad to the more specific domain (Reyes, Caicedo, and Camargo 2015).

This is exactly the approach taken in this study: a first model is trained on a large Wikipedia dataset, and the parameters are transfered from the broad domain to the specific domain, which, in this case, is American poetry. We then fine-tune our model using early stopping and a small learning rate. Moreover, we remove the original softmax layer and replace it with a new one. The new layer is trained from scratch using back-propagation with data from the poetry corpus. The training batch size is set to 32, the number of iterations to 10, and the learning rate used is 0.0005.

Experiments

The aim of the experiments is to evaluate to what extent the neural language model can learn to predict concrete and abstract entities in American modernist poetry, especially across authors; or whether such patterns are entirely idiosyncratic and, hence, unpredictable.

Data

The dataset crawled from PoetryDB has been divided into 80% training, 10% development, and 10% test set. For our test set, we check if the target nouns have more than five synonyms in the Princeton Wordnet (Miller 1995; Fellbaum 1998). The average number of synonyms per target word is 7.55.

In our experiments, we either use synonyms or words of the same semantic category as alternative candidate words. All of these are retrieved from Princeton WordNet (Miller 1995; Fellbaum 1998).[5] Often the candidate words differ in register, i.e., degree of formality, such as in:

EXAMPLE:	To that new ___			
CANDIDATES:	matrimony	union	marriage	wedlock

In other cases, especially when the candidate words are simply instances of the same semantic category, the differences are of a semantic nature:

[2] [https://github.com/FiammettaC/Predicting-Concrete-and-Abstract-Entities-in-Modern-Poetry]

[3] Anonymized.

[4] https://corpus.byu.edu/wiki/. Accessed [10-02-2018].

[5] https://wordnet.princeton.edu/

EXAMPLE:	Shaking his ___, as in doubt; then, heaving a sigh, ...
CANDIDATES:	hand mind head leg

Performance metrics

The first two experiments are evaluated by calculating the Mean Reciprocal Rank (MRR), i.e., the harmonic mean of the ranks of the true answer, that is, the removed target word. We use this performance metric, since our task is to reconstruct a missing noun in each verse, and there is only one relevant answer; hence, Mean Average Precision (MAP) is not applicable, for example. MRR, however, also rewards predicting the target word as the second most probable candidate, for example, which we believe is fair, since the candidate list may include other poetic entities that the poet did not even consider. In our third experiment, in which we compare the results of the model with the annotations of actual, professional poets, we also include a simple accuracy metric.

Vocabularies

In the first experiment, the vocabulary is restricted to synonyms; in our second experiment, to words of the same semantic category. In the last experiment, where we compare our model's performance to professional poets, we restricted the number of candidates to four, which were either synonyms of the target noun or semantically related words.

The semantically related words were retrieved in the following way: The hypernym of each target noun was retrieved, and we then retrieved all its hyponyms and made them possible candidates.

Technically, we restrict the output vocabulary by (pairwise) multiplying the softmax output by a binary indicator vector encoding the words in our current vocabulary. We then sort the probabilities assigned to all candidate words to obtain a ranking and calculate the MRR of our predictions.

Results

Experiment I and II

Table 1 lists the results of Experiments I and II. Our first observation is that there is a very significant improvement in the performance of the fine-tuned language model, compared to the baseline, across the board. The error reduction in Experiment I, for example, is 34%.

Our second and perhaps more important observation is this: Comparing the results of the first experiment on randomly selected instances to our results on unseen data shows, unsurprisingly, a slight decrease in performance. This is probably the result of idiosyncratic differences between the various authors in our dataset. Note, however, that training on other poets still enables our fine-tuned language model to learn something general about poetic language and better predict entities referred to across poets. The error reduction over our baseline is still 14.1% in Experiment 1, when evaluating across poets. We thus conclude that the *labor limae* of poets, i.e, their lexical preferences are not entirely idiosyncratic, but to some extent learnable from data.

We even performed an experiment on held-out British poets, not reported here, where we observed similar results to Experiment II. This could suggest that the general poetic preferences modeled in Experiment II are not very sensitive to cultural differences.

Experiment III

Experiment III is the comparison of our model against professional poets. It was not possible to calculate the MRR scores on the human annotations, since the task given to the poets was to annotate which noun would fit better into the verse, and not to sort the given options based on what they thought the appropriate word was. We will compute the MRR of the model, for comparison with Experiment I and II, but the performance metric used to compare the model and the human annotators will be simple accuracy scores.

Table 2 shows that there is a large improvement in the performance of the fine-tuned model, compared to the pretrained baseline model. Although the results are significantly higher than in the other experiments, it is important to bear in mind that the in this settings we only need to consider four candidates, while in the other experiments, we were presented with 7.5 candidates on average.

As already mentioned, the inter-annotator agreement was calculated using Fleiss' Kappa. The Kappa value in the annotations is: 0.332. This value of kappa is quite low, highlighting the difficulty of the task and the idiosyncracies of professional poets. Table 3 furthermore shows that the candidates presented to the professional poets were quite plausible, since the four annotators have an average accuracy of only 35%. This means that *our language model outperforms all of the professional poets*. The accuracy of the pretrained model, on the other hand, is lower than the results obtained by the poets, meaning that only training the model on a Wikipedia Corpus did not provide the network with necessary knowledge to predict the correct entities in modern poetry.

Analysis

Our first observation analyzing the data and the performance of our models is that the synonyms and the semantically related entities make for very different candidate sets. See, e.g., the example below:

TARGET WORD:	love
SYNONYMS:	['honey', 'dear', 'passion', 'dearest', 'beloved', 'love']
HYPER-HYPONYMS:	['anxiety', 'hate', 'ire', 'veneration', 'fear', 'anger', 'love']

The synonyms appear to be more pertinent to the target noun, while the nouns in the hypernyms-hyponyms list are more general. They are related to the same field of the target word, but (often) not to the word directly. It is also clear in this example that, while the synonyms all refer to a person,

	EXPERIMENT I		EXPERIMENT II	
Test Set	Randomly selected	Unseen Authors	Randomly selected	Unseen Authors
MRR Pre-Trained Model	0.435	0.433	0.397	0.424
MRR Fine-Tuned Model	0.627	0.513	0.518	0.426
Error Reduction in %	34.0	14.1	20.0	0.3

Table 1: Results of Experiments I and II

EXPERIMENT III	
MRR Pre-trained Model	0.535
MRR Fine-Tuned Model	0.640
Error Reduction in %	22.6

Table 2: MRR results of Experiment III

the words on the hypernyms-hyponyms list refer to abstract entities. We were interested in whether the fine-tuned language model improved more over our baseline with concrete or abstract entities, and we will use the rest of the paper to analyze the performance of the language models at the level of such semantic categories.

Semantic categories

First we group our predictions into four:

- Entities that have been correctly classified by both models;
- Entities that have been incorrectly classified by both models;
- Entities that have been correctly classified only by the pre-trained model;
- Entities that have been correctly classified only by the fine-tuned model.

We will now analyze the distribution of WordNet semantic categories over these four groups of entities. First we consider the very coarse-grained distinction between concrete and abstract entities alluded to several times in this paper. These correspond to the WordNet super-classes of *abstraction* and *physical entity*. These two categories are defined in WordNet as:

- **Abstraction:** a concept or idea not associated with any specific instance;
- **Physical Entity:** an entity that has physical existence.

Our second analysis goes one step deeper in the WordNet ontology, looking at third-level semantic categories. The third level contains the following categories:

- **Process:** a particular course of action intended to achieve a result (e.g. energy, dissolution, breeze, light, tears, vision, sense, breath, wind);
- **Causal Agent:** any entity that produces an effect or is responsible for events or results (e.g. friends, ottoman, hero, sailors, father, king, souls);
- **Matter:** a vaguely specified concern (e.g. clay, air, blood, sand, wood, silver, water, particle, crystal);
- **Abstract Entity:** a general concept formed by extracting common features from specific examples;
- **Object:** a tangible and visible entity; an entity that can cast a shadow;
- **Thing:** a special situation (e.g. body parts).

Figure 1 shows the error reductions of our fine-tuned language model over our baseline, for concrete and abstract entities, in Experiment I. The main observation, which is one of the important findings of this study, is that a large fraction of the improvements we obtain from fine-tuning our language model, comes from concrete entities. Since many people associate poetry with relatively abstract language, this may be surprising, but especially in modernist poetry, one key characteristic is often the *novel use of concrete entities in new contexts*. This is exactly the kind of language use that would challenge our baseline model; our fine-tuned neural language model seems to have adapted to many of these new associations of concrete entities with alternative contexts.

Figure 1 also shows the corresponding numbers for coarse-grained semantic categories, corresponding to the third level of the Princeton Wordnet ontology. Some semantic categories being slightly more predictable than others. The coarse-grained semantic class MATTER, we can predict with a 10% error rate, for example. This is in part due to a ¿35% error reduction when fine-tuning our model on poetry data. The fine-tuned neural language model improves even more over our baseline when predicting THINGS. In general, the models are challenged when predicting entities that are processes.

Discussion

Our findings in this paper were, first and foremost: (a) It is possible to fine-tune a language model to predict entities in modernist poetry, even across poets. Importantly, this means that the introduction of entities is not entirely idiosyncratic or unpredictable. A fine-tuned language model is considerable better, however, than a language model trained on ordinary language, but we also show that (b) most of the error reduction from fine-tuning is from improved performance on predicting *concrete* entities – *things*, in particular – which highlights how a hallmark of modernist poetry is *not how abstract entities are discussed, but how concrete entities are used in new contexts*. Finally, we showed that our fine-tuned language model was even better at predicting these entities than professional poets, suggesting that our model succesfully learned to identify poetic entities, and that most of the remaining variance is idiosyncratic.

There is one other dimension that we need to take into account: the role of metaphors. In modern poetry, *concrete*

EXPERIMENT III						
	Annotator 1	**Annotator 2**	**Annotator 3**	**Annotator 4**	**Pre-trained Model**	**Fine-tuned Model**
Accuracy %	38.7	30.7	37.3	33.3	26.7	40.0

Table 3: Accuracy results of Experiment III; comparison with professional poets

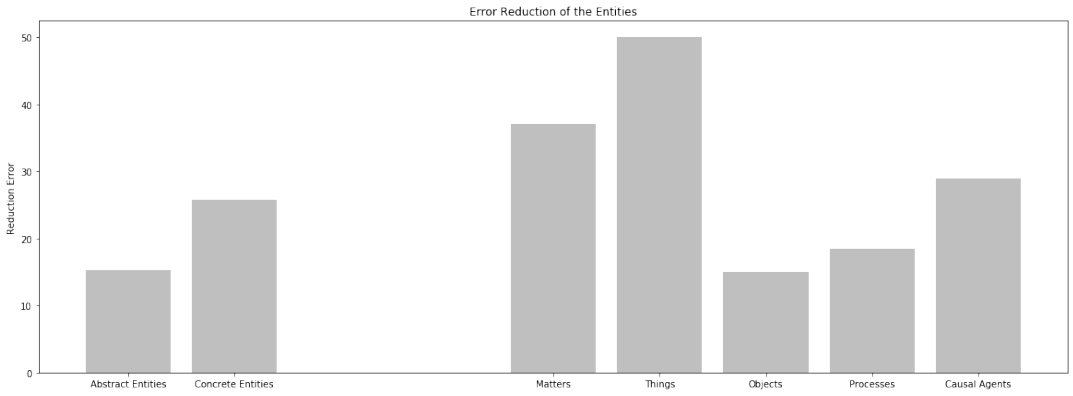

Figure 1: Error reductions for abstract and concrete entities (left) and for coarse-grained semantic classes (right). See Experiment I.

objects are often used to refer to something *else*, something more abstract: Concrete entities are evocative, not only alluding to the objects they refer to, but also to another, higher or more abstract meaning. One example of this is what T. S. Eliot called the *objective correlative*; a term he used to describe "a set of objects, a situation, a chain of events which shall be the formula of that particular emotion" that the poet feels and hopes to evoke in the reader. Also in metaphors, the *concrete words* are used in new contexts. Whether the concrete words are introduced in new contexts to create poetic effects through metaphor or by other means, however, is not a distinction that we try to draw in this paper.

Conclusion

In this study, we showed that it is possible to fine-tune a language model to predict entities in modernist poetry, even across poets. Using various set-ups, our language model was able to predict the correct entities from sets of (at least five) synonyms or semantically related words with an accuracy of 40% or more, and we showed that our model even was able to outperform professional poets in this task. Importantly, this means that the introduction of entities in modernist poetry is not entirely idiosyncratic or unpredictable. A fine-tuned language model is considerable better, however, than a language model trained on ordinary language. We also show that most of the error reduction from fine-tuning is from improved performance on predicting *concrete* entities – *things*, in particular – which highlights how a hallmark of modernist poetry is *not how abstract entities are discussed, but how concrete entities are used in new contexts*.

References

Bernstein, C. 1987. *Artifice of Absorption*. Singing Horse Press.

Fabb, N. 2006. Generated metrical form and implied metrical form nigel fabb. *Formal Approaches to Poetry: Recent Developments in Metrics* 11:77.

Fang, A. C.; Lo, F.; and Chinn, C. K. 2009. Adapting nlp and corpus analysis techniques to structured imagery analysis in classical chinese poetry. In *Proceedings of the Workshop on Adaptation of Language Resources and Technology to New Domains*, 27–34. Association for Computational Linguistics.

Felbo, B.; Mislove, A.; Søgaard, A.; Rahwan, I.; and Lehmann, S. 2017. Using millions of emoji occurrences to pretrain any-domain models for detecting emotion, sentiment, and sarcasm. In *EMNLP*.

Fellbaum, C. 1998. A semantic network of english verbs. *WordNet: An electronic lexical database* 3:153–178.

Forstall, C. W.; Jacobson, S. L.; and Scheirer, W. J. 2011. Evidence of intertextuality: investigating paul the deacon's angustae vitae. *Literary and Linguistic Computing* 26(3):285–296.

Genzel, D.; Uszkoreit, J.; and Och, F. 2010. Poetic statistical machine translation: rhyme and meter. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 158–166. Association for Computational Linguistics.

Ghazvininejad, M.; Shi, X.; Choi, Y.; and Knight, K. 2016. Generating topical poetry. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 1183–1191.

Girosi, F.; Jones, M.; and Poggio, T. 1995. Regularization theory and neural networks architectures. *Neural Computation* 7:219–269.

Greene, E.; Bodrumlu, T.; and Knight, K. 2010. Automatic analysis of rhythmic poetry with applications to gen-

eration and translation. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, 524–533. Association for Computational Linguistics.

Hayward, M. 1996. Analysis of a corpus of poetry by a connectionist model of poetic meter. *Poetics* 24(1)(1–11.).

He, Z.-S.; Liang, W.-T.; Li, L.-Y.; and Tian, Y.-F. 2007. Svm-based classification method for poetry style. In *Machine Learning and Cybernetics, 2007 International Conference on*, volume 5, 2936–2940. IEEE.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Howard, J., and Ruder, S. 2018. Universal language model fine-tuning for text classification. In *ACL*.

Java, J. 2015. *Characterization of Prose by Rhetorical Structure for Machine Learning Classification*. Ph.D. Dissertation, Nova Southeastern University.

Kao, J., and Jurafsky, D. 2012. A computational analysis of style, affect, and imagery in contemporary poetry. In *Proceedings of the NAACL-HLT 2012 Workshop on Computational Linguistics for Literature*, 8–17.

Kaplan, D. M., and Blei, D. M. 2007. A computational approach to style in american poetry. In *icdm*, 553–558. IEEE.

Kesarwani, V. 2018. *Automatic Poetry Classification Using Natural Language Processing*. Ph.D. Dissertation, Université d'Ottawa/University of Ottawa.

L'Hostis, G.; Grangier, D.; and Auli, M. 2016. Vocabulary selection strategies for neural machine translation. *arXiv preprint arXiv:1610.00072*.

Luong, M.-T.; Kayser, M.; and Manning, C. 2015. Deep neural language models for machine translation. In *CoNLL*.

Manurung, R.; Ritchie, G.; and Thompson, H. 2012. Using genetic algorithms to create meaningful poetic text. *Journal of Experimental & Theoretical Artificial Intelligence* 24(1):43–64.

McCloskey, M., and Cohen, N. J. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. *The Psychology of Learning and Motivation* 24:109–165.

Mi, H.; Wang, Z.; and Ittycheriah, A. 2016. Vocabulary manipulation for neural machine translation. *arXiv preprint arXiv:1605.03209*.

Miller, G. A. 1995. Wordnet: a lexical database for english. *Communications of the ACM* 38(11):39–41.

Mocialov, B.; Turner, G.; and Hastie, H. 2018. Transfer learning for british sign language modelling. In *COLING Workshop on NLP for Similar Languages, Varieties and Dialects*.

Pan, S. J.; Yang, Q.; et al. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22(10):1345–1359.

PoetryDB. Poetrydb api.

Reyes, A. K.; Caicedo, J. C.; and Camargo, J. E. 2015. Fine-tuning deep convolutional networks for plant recognition. In *CLEF (Working Notes)*.

Sennrich, R.; Haddow, B.; and Birch, A. 2016. Improving neural machine translation models with monolingual data. In *ACL*.

Valéry, P. 1957. *Œuvres de Paul Valéry*. Paris.

Zhang, X., and Lapata, M. 2014. Chinese poetry generation with recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 670–680.