Poetry of the Crowd: A Human Computation Algorithm to Convert Prose into Rhyming Verse

Quanze Chen, Chenyang Lei, Wei Xu, Ellie Pavlick and **Chris Callison-Burch**
Computer and Information Science Department
University of Pennsylvania, Philadelphia, PA, USA
{cquanze, chenylei, xwe, epavlick, ccb}@seas.upenn.edu

Abstract

Poetry composition is a very complex task that requires a poet to satisfy multiple constraints concurrently. We believe that the task can be augmented by combining the creative abilities of humans with computational algorithms that efficiently constrain and permute available choices. We present a hybrid method for generating poetry from prose that combines crowdsourcing with natural language processing (NLP) machinery. We test the ability of crowd workers to accomplish the technically challenging and creative task of composing poems.

Introduction

Automatic poetry generation is a fascinating natural language generation challenge that has been attempted by many researchers and engineers in the past. Most previous work has approached the problem as purely mechanical, either focusing on modern poetry generation from random words or on a specific subproblem like meter analysis (Greene, Bodrumlu, and Knight 2010). Generating poems that are well structured and meaningful remains an unsolved problem. In this research we present a **human computation algorithm** for **generating poetry from prose**. We test the capability of crowdsourcing to accomplish this complex creative activity with the aid of NLP algorithms for expanding lexical choice, identifying rhyming words, and calculating meter.

We contend that for tasks like poetry generation, humans excel in various aspects like picking contextually relevant phrase substitutions and composing grammatically correct sentences from a suggested set of words. Computers excel at other dimensions. Existing algorithms and datasets can suggest potential paraphrases, or calculate the meter of a verse, or list rhyming words, but computers have trouble constructing sensible sentences. We can take advantage of the skills of both humans and computers by splitting our task into a series of subtasks (sketched in Figure 1).

Our Approach

One approach to generating poetry from prose is to consider the problem is two-phased. The first phase is the collection, or brainstorming, of a pool of valid alternatives for each of the expressions in the prose. This creates the set of phrase variations we build our poetry from. The second phase is assembly of the output poetry under the required constraints using choices from the first phase. In the case of rhyming couplets, we will focus on the meter and rhyme constraints. For each task we are able to make effective use of machines and humans.

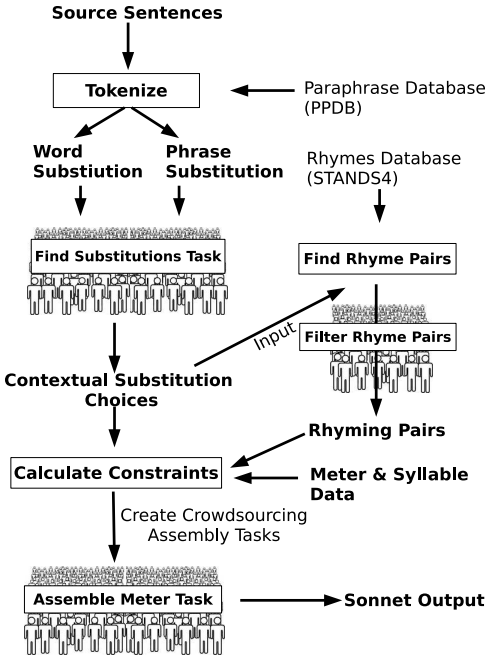

Figure 1: We combine the creative powers and language skills of people with computational algorithms for expanding paraphrase, identifying rhymes and calculating meter

Lexical Expansion The first step in our process is to expand all the words and phrases in the input prose with meaning-equivalent alternatives. We make use of the Paraphrase Database (PPDB), an automatically constructed collection of 120 million meaning-equivalent re-write rules (Ganitkevitch, VanDurme, and Callison-Burch 2013). A naive application of PPDB paraphrases is not reliable. The programmatically constructed PPDB contains some noise

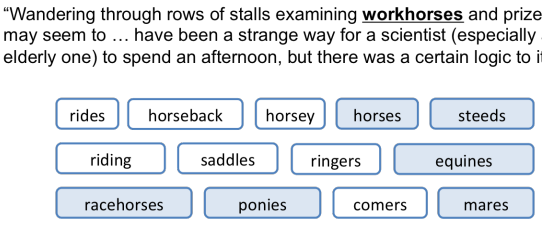

Figure 2: Appropriate lexical substitutions (in shaded boxes) selected by crowdsourcing workers regarding to the context.

and incorrect paraphrases. Moreover, many of the paraphrases are correct only in particular contexts (McCarthy and Navigli 2007). However, we can enhance the quality of our word expansion by having crowd workers filter for contextually appropriate paraphrases. Workers are shown the sentence context and asked to select context fitting paraphrases. We take a majority vote from 5 workers' judgments about each paraphrase. Paraphrases that receive 3 or more votes stay in the substitution set. Figure 2 shows an example.

Meter Different poem types have different meter constraints. For example, a sonnet should follow iambic pentameter constraints, implying 5 weak-strong syllable pairs per line. It is often difficult for untrained crowd workers to perform scansion on a large set of choice words. Providing pre-calculated meter greatly simplifies this task. We obtain the stress pattern for each word by combining lookups in the CMU pronunciation dictionary (Bartlett, Kondrak, and Cherry 2009) with analysis of predicted meter probability results from a machine learning algorithm trained on sonnet data (Greene, Bodrumlu, and Knight 2010). If a sufficiently reliable result cannot be found from these two sources, we approximate the meter by programmatically computing the syllable count. We use this procedure to collect all possible meter patterns for each alternative choice generated in the expansion step.

Rhyme Rhyme information is acquired through the STANDS4 API, an online dictionary service.[1] We collect rhyme information of each word encountered in the original text or expanded word set. The intersection of the rhyming sets across different words is taken to find rhyming pairs of words. Using the expanded word set greatly improves our chances of finding valid matching rhyme pairs. In the preliminary experiment, we use the first 5 paragraphs from the book *The Wisdom of Crowds* by James Surowiecki as the input text. We were able to find 179 rhyming pairs (82 unique words) in the original prose, however, with the expanded word set we are able to increase this number to 911 rhyming pairs (295 unique words).

Writing Iambic Pentameter We ask crowd workers to write lines of poetry based on the input prose and expanded alternative word choices. The worker is presented with an interface showing the original sentence and available word set along with all meter information, as shown in Figure 3. An algorithm dynamically compares the composed sentence and the alternative words set to an iambic pentameter template. The worker can build a sentence with the guarantee that available words always fit the meter constraint. To guarantee rhyme, we constrain the ending word or phrase to be one drawn from a rhyming pair in word sets across different sentences and adjust the algorithm template to omit the ending meter accordingly. We rely on the workers to explore the search space and assemble sensible sentence paths capturing aspects of the input prose.

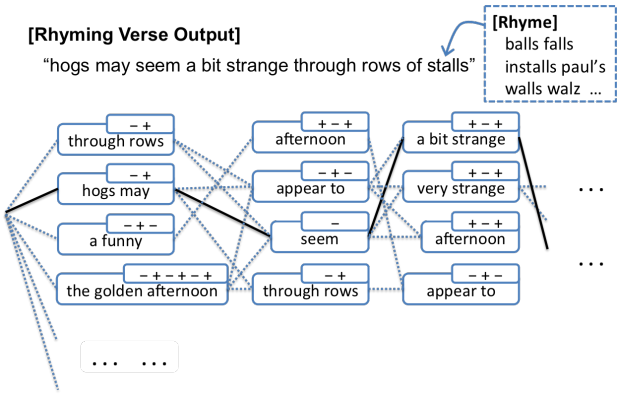

Figure 3: A crowd worker composes a sentence path with the "- + - + - + - + - +" stress pattern (iambic pentameter). The ending word *stalls* rhymes with other words that terminate sentences constructed in another worker's task.

References

Bartlett, S.; Kondrak, G.; and Cherry, C. 2009. On the syllabification of phonemes. In *Proceedings of NAACL*.

Ganitkevitch, J.; VanDurme, B.; and Callison-Burch, C. 2013. PPDB: The paraphrase database. In *Proceedings of NAACL*.

Greene, E.; Bodrumlu, T.; and Knight, K. 2010. Automatic analysis of rhythmic poetry with applications to generation and translation. In *Proceedings of EMNLP*.

McCarthy, D., and Navigli, R. 2007. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the 4th International Workshop on Semantic Evaluations*.

Surowiecki, J. 2005. *The Wisdom of Crowds*. Random House LLC.

[1] http://www.rhymes.net/rhymes_api.php