Generating similes ~~effortlessly~~ *like a Pro*: A Style Transfer Approach for Simile Generation

Tuhin Chakrabarty[1,2]*, **Smaranda Muresan**[2,4] **and Nanyun Peng**[1,3]
[1]Information Sciences Institute, University of Southern California
[2]Department of Computer Science, Columbia University
[3]Computer Science Department, University of California, Los Angeles
[4]Data Science Institute, Columbia University

{tuhin.chakr, smara}@cs.columbia.edu
violetpeng@cs.ucla.edu

Abstract

Literary tropes, from poetry to stories, are at the crux of human imagination and communication. Figurative language, such as a simile, goes beyond plain expressions to give readers new insights and inspirations. We tackle the problem of simile generation. Generating a simile requires proper understanding for effective mapping of properties between two concepts. To this end, we first propose a method to automatically construct a parallel corpus by transforming a large number of similes collected from Reddit to their literal counterpart using structured common sense knowledge. We then fine-tune a pretrained sequence to sequence model, BART (Lewis et al., 2019), on the literal-simile pairs to generate novel similes given a literal sentence. Experiments show that our approach generates 88% novel similes that do not share properties with the training data. Human evaluation on an independent set of literal statements shows that our model generates similes better than two literary experts *37%*[1] of the times, and three baseline systems including a recent metaphor generation model *71%*[2] of the times when compared pairwise.[3] We also show how replacing literal sentences with similes from our best model in machine generated stories improves evocativeness and leads to better acceptance by human judges.

1 Introduction

Comparisons are inherent linguistic devices that express the likeness of two entities, concepts or ideas. When used in a figurative sense, these comparisons are called similes. They are a figure of speech that compare two different kind of things, usually with the intent to make the description more emphatic or vivid, being often used in literature and poetry to spark the reader's imagination (Paul et al., 1970). Take the following two examples: "The city was *like a painting*", and "If it falls into the wrong hands it would be as catastrophic *as a nuclear bomb*." In the first example, the comparison draws on the implicit "beauty" property being shared by the two very different entities, *city* and *painting*, while in the second the "catastrophic" property is shared by *falling into the wrong hands* and *nuclear bomb*.

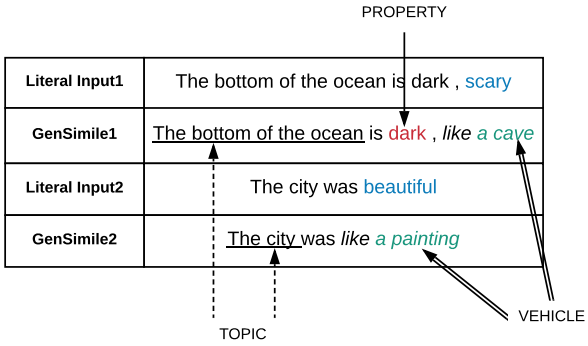

Figure 1: Examples of two generated similes GenSimile1 and GenSimile2 from their literal inputs.

While most computational work has focused on simile detection (Niculae and Danescu-Niculescu-Mizil, 2014; Mpouli, 2017; Qadir et al., 2015, 2016; Zeng et al., 2019; Liu et al., 2018), research on simile generation is under-explored. Generating similes could impact many downstream applications such as creative writing assistance, and literary or poetic content creation. To tackle the generation problem, we take advantage of the relatively simple structure of similes that consists of five elements (Hanks, 2013; Niculae and Danescu-Niculescu-Mizil, 2014): the `TOPIC` (usually a noun phrase that acts as the logical subject), the `VEHICLE` (the logical object of the comparison, usually a noun phrase), the `PROPERTY` (what

* The research was conducted when the author was at USC/ISI.

[1]We average 32.6% and 41.3% for 2 humans.

[2]We average 82% ,63% and 68% for three baselines.

[3]The simile in the title is generated by our best model. Input: Generating similes effortlessly, output: Generating similes *like a Pro*.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 6455–6469,
November 16–20, 2020.

the two things being compared have in common, usually an adjective), the EVENT (eventuality or state, usually a verb), and the COMPARATOR (the trigger word or phrase that marks the presence of a comparison, usually the preposition "like" or "as...as"). All elements of a simile are explicit, with the exception of PROPERTY, which can be both implicit and explicit. If we take the first example above, its structure is: "[The city/TOPIC] [was/EVENT] [like/COMPARATOR] [a painting/VEHICLE]" (PROPERTY is implicit). Unlike metaphors, the semantic context of similes tends to be very shallow, transferring a single *property* (Hanks, 2013). Moreover, the explicit syntactic structure of similes allows, in exchange, for more lexical creativity (Niculae and Danescu-Niculescu-Mizil, 2014).

We focus on the task of generating a simile starting from a literal utterance that contains the TOPIC, EVENT and PROPERTY. We frame this task as a style-transfer problem (Shen et al., 2017; Fu et al., 2017; Li et al., 2018; Sudhakar et al., 2019), where the author's intent is to make the description of the TOPIC more emphatic by introducing a comparison with the VEHICLE via a shared PROPERTY (See Figure 1 for examples of literal descriptive sentences and the generated similes). We call our approach **SCOPE** (**S**tyle transfer through **CO**mmonsense **P**rop**E**rty). There are two main challenges we need to address: 1) the lack of training data that consists of pairs of literal utterances and their equivalent simile in order to train a supervised model; 2) ensuring that the generated simile makes a meaningful comparison between the TOPIC and the VEHICLE via the shared PROPERTY explicitly or implicitly expressed (e.g., Figure 1 GenSimile1 and GenSimile2, respectively). To the best of our knowledge, this is the first work in attempting to generate similes. By framing the task as a style-transfer problem we make three contributions: [4]

Automatic creation of a parallel corpus of *[literal sentence, simile]* pairs. Our constructed corpus contains 87,843 such pairs. As a first step, we use distant supervision to automatically collect a set of *self-labeled similes* using the phrase *like a*. We then convert these similes to their literal versions by removing the COMPARATOR and replacing the VEHICLE with the associated PROPERTY by leveraging the structured common sense knowledge achieved from COMET (Bosselut et al., 2019), a language model fine-tuned on ConceptNet (Speer et al., 2017). For example, for the simile "Love is like a unicorn" our method will generate "Love is rare" (Section 2.1).

Transfer learning from a pre-trained model for generating high quality similes. Our system **SCOPE**, *fine-tunes* BART (Lewis et al., 2019) — a state of the art pre-trained denoising autoencoder built with a sequence to sequence model, on our *automatically collected parallel corpus* of *[literal sentence, simile]* pairs (Section 2.2) to generate similes. Human evaluations show that this approach generates similes that are better 37% of the time on average compared to 2 literary experts, 82% and 63% of times compared to two well-crafted baselines, and 68% of the times compared to a state of the art system for metaphor generation (Stowe et al., 2020) (Section 4).

A task-based evaluation. We show the effectiveness of the generated similes as a tool for enhancing creativity and evocativeness in machine generated stories. Evaluation via Amazon Mechanical Turk shows that stories containing similes generated by **SCOPE** is preferred by Turkers 42% of the times compared to stories without similes, which is preferred 25% of the times (Section 6).

2 SCOPE: Style Transfer through COmmonsense PropErty

Our style transfer approach for simile generation from literal descriptive sentences has two steps: 1) first convert self-labeled similes into literal sentences using structured common sense knowledge (Section 2.1); and 2) given the *[literal sentence, simile]* pairs, fine-tune a seq2seq model on these pairs to generate a simile given a literal sentence (Section 2.2). This two-step approach is shown in the upper half of Figure 2.

2.1 Automatic Parallel Corpus Creation

One of the requirements to train a supervised generative model for text style transfer is the presence of a large-scale parallel corpus. We use distant supervision to collect self-labeled similes using the phrase *like a* from Reddit (e.g., the rows labeled as Simile in Table 1). For fine-tuning, the similes form the "target" side of our parallel data. For the "source" side of our parallel data, we use commonsense knowledge to transform the similes to their

[4] Code & Data at https://github.com/tuhinjubcse/SimileGeneration-EMNLP2020

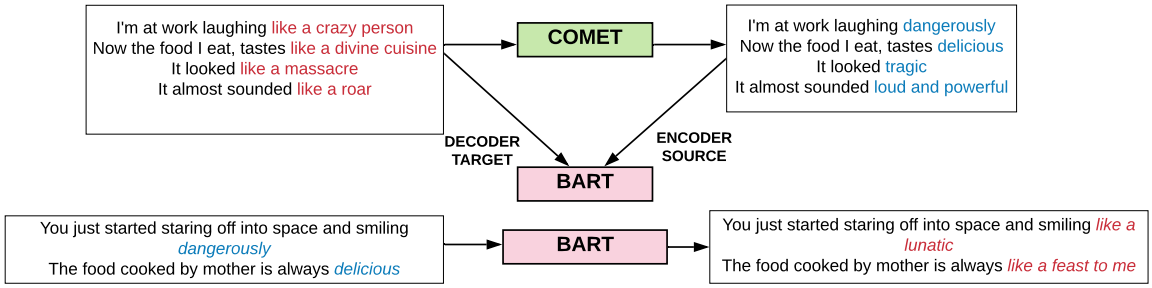

Figure 2: A schematic illustration of our system, where the top block shows our **training** process where we use COMET to transform similes to literal sentences and use them to fine-tune BART. The block below shows the **inference** step where we use fine-tuned BART to generate novel similes conditioned on a literal sentence.

literal version (e.g., the rows labeled as Best Literal in Table 1).

Simile Dataset Collection. One of the possible ways to collect similes would be to train a supervised model using existing data and methods for simile detection but most data sets are very small in size (in the order of a few hundreds). The only large-scale dataset is that of (Niculae and Danescu-Niculescu-Mizil, 2014), however their data is from a rather restricted domain of product reviews on Amazon, which might lack variety, diversity and creativity needed for this task. For our work, we hypothesize that similes are used frequently in creative writing or humorous content on social media (Veale, 2013). Hence, we obtain training data by scraping the subreddits WRITINGPROMPTS [5] and FUNNY [6] from social media site Reddit for comments containing the phrase *like a*. Similes can be both Open and Closed. For example the Closed Simile, *"The boy was as strong as an ox"* gives **strong** as the PROPERTY shared by the *boy* and *ox*. But most similes do not give an explicit PROPERTY such as the Open Simile (e.g., *"The boy was like an ox")* leaving the reader to infer that the boy is strong/large/fast (Qadir et al., 2016). Due to their implicit nature, generating open similes is often more challenging and hence we resort to only using *like a*[7] as a comparator instead of *as...as*. We use the API provided by pushshift.io [8] to mine comments. Through this process we collect 87,843 self-labeled human written similes, from which we use 82,697 samples for training and 5,146 for validation.

Simile	Love is like a *unicorn.*
Has property	very rare, rare, beautiful, beautiful and smart, color
Best Literal	Love is *rare.*
Simile	It was cool and quiet, and I stormed through like a *charging bull.*
Has property	big and strong, dangerous, big, fast, large
Best Literal	It was cool and quiet, and I stormed through *fast.*
Simile	Sir Francis's voice was calm and quiet, like a *breeze through a forest.*
Has property	very relax, soothe, cool, beautiful, relax
Best Literal	Sir Francis's voice was calm and quiet, *very relaxed.*

Table 1: Examples of self-labeled similes collected from Reddit. For each example, we show the top five commonsense properties associated with the *vehicle* obtained from COMET, and the best literal sentence constructed from these properties. The blue italic texts in the literal sentences represent the *property* inferred from the *vehicle* in the simile (denoted in black italic).

Simile to Literal Transformation via Commonsense Property. From a theoretical perspective, similes are created by making a comparison between the TOPIC and the VEHICLE through a shared PROPERTY. While this property is naturally known to humans through common sense and connotative knowledge, computers still struggle to perform well on such tasks when the PROPERTY is not expressed. Hence we use structured common sense knowledge to derive properties to transform similes to their literal versions.

To generate the common sense PROPERTY that is implied by the VEHICLE in the simile, we take advantage of the simple syntactic structure of a simile. We extract the VEHICLE by extract-

[5] https://www.reddit.com/r/WritingPrompts/

[6] https://www.reddit.com/r/funny/

[7] While there can be noisy sentences where *like a* does not introduce a simile (e.g., the TOPIC is a PP and the sentence is <= 6 tokens such as *I feel like a .., I would like a .., I don't like a..*), these instances are rare (1.1 %), so we do not remove them. More details in Appendix A.2

[8] https://pushshift.io/

ing the phrase after *like a* and feed it as input to COMET (Bosselut et al., 2019). COMET is an adaptation framework for constructing commonsense knowledge based on pre-trained language models. Our work only leverages the **HasProperty** relation from COMET [9].

For a given simile *'Love is like a unicorn.'*, the TOPIC *Love* is compared to the VEHICLE *unicorn*. As shown in Table 1, COMET tells us the top 5 properties associated with the VEHICLE are *very rare, rare, beautiful, beautiful and smart, color*. COMET gives us the properties sorted by probability in isolation by just relying on the VEHICLE. While in most situations all of the properties are apt, we need to make the literal sentence as meaningful as possible. To do this, we append the common sense property to the portion of the simile before *'like a'*. This typically consists of the TOPIC, the EVENT, and a PROPERTY if stated explicitly. We take the top 5 properties from COMET to form 5 possible literal versions for a particular simile. To rank these literal versions and select the best one, we rely on perplexity scores obtained from a pre-trained language model GPT (Radford et al., 2018). Table 1 shows human written similes collected from Reddit, the top 5 common sense properties associated with the VEHICLE, and the literal version created by taking the best PROPERTY. To correct any grammatical errors introduced by this manipulation, we rely on a grammatical error correction model (Zhao et al., 2019).

Test Data Collection. Our task is to generate a simile given a literal input. The automatically-generated parallel data might contain stylistic biases. To truly measure the effectiveness of our approach, we need to evaluate on a dataset independent of our training and validation data. Towards this end, we again scrape WRITINGPROMPTS subreddits for sentences which are this time *literal* in nature (without any comparators *like, as*). Since literal utterances contains the description of TOPIC via a PROPERTY and usually the PROPERTY is an adjective or adverb, we restrict the last word of our literal sentences to adverbs or adjectives. We crawl 500 such sentences and randomly sample 150 literal utterance. We used two literary experts (not authors of this paper) — a student in creative writing, and a student in comparative literature who is the author of a novel — to write corresponding similes for each of these 150 inputs for evaluation and comparison.

[9] https://mosaickg.apps.allenai.org/comet_conceptnet

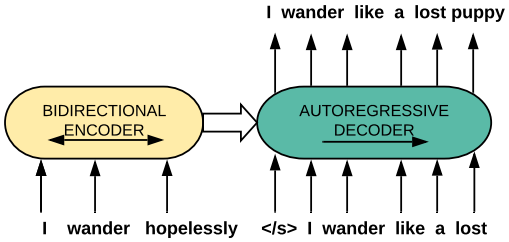

Figure 3: The backbone of SCOPE: fine-tuning BART on literal to simile pairs.

2.2 Seq2Seq Model for Simile Generation

Our goal of generating similes can be broken down into two primary tasks: 1) identifying the words in the literal sentence that should be removed or replaced and 2) generating the appropriate substitutions while being pertinent to the context. Sequence to sequence (seq2seq) neural network models (Sutskever et al., 2014) have demonstrated great success in many text generation tasks, such as machine translation, dialog system and image caption, with the requirement of a considerable amount of parallel data. Hence, we use seq2seq models for simile generation.

BART (Lewis et al., 2019) is a pre-trained model combining bidirectional and auto-regressive transformers. It is implemented as a sequence-to-sequence model with a bidirectional encoder over corrupted text and a left-to-right autoregressive decoder. In principle, the pre-training procedure has two stages: (1) text is corrupted with an arbitrary noising function, and (2) a transformer-to-transformer model is learned to reconstruct the original text. Because BART has an autoregressive decoder, it can be directly fine-tuned for most sequence generation tasks. Here, the encoder input is a sequence of words, and the decoder generates outputs autoregressively, as shown in Figure 3. BART achieves new state-of-the art results on a number of text generation tasks, making it an ideal choice for generating similes. We refer the reader to (Lewis et al., 2019) for further details.

For our task, we fine-tune BART by treating the literal input as encoder source and the simile as the the decoder target. Post fine-tuning at the inference step, we use top-k sampling strategy (Fan et al., 2018) to generate similes conditioned on a test literal input.

Implementation details. Hyper-parameters, and essential details needed for reproducing experiments are given in Appendix A.1.

3 Experimental Setup

To compare the quality of the generated similes, we benchmark our SCOPE model against human performance (i.e., the two creative writing experts HUMAN1 & HUMAN2 described in Section 2.1) and three baseline systems described below

3.1 Baseline Systems

Simile generation is a new task. The baselines outlined below have been used for other generation tasks. We adapt them to generate similes.

1. **BART**: This is the pre-trained BART model. Since BART is a pre-trained sequence to sequence model, it can still be used for conditional text generation. To this end we use the same literal sentence (For example *The city was beautiful*) as an input to the encoder and force the decoder to begin with same prefix by removing the adjective/adverb at the end and appending the comparator and the article (*The city was like a*) and generate a simile.

2. **Retrieval (RTRVL)**: We also experiment with a retrieval approach where we retrieve a VEHICLE from ConceptNet (Speer et al., 2017) having the highest *HasProperty* relation w.r.t our input (i.e., an adjective or adverb at the end of literal sentence)[10]. For the input *The city was beautiful* we query ConceptNet with *beautiful*, which returns *sunset* as the VEHICLE having the highest weight for *HasProperty beautiful*. We take this retrieved VEHICLE and append it to the prefix ending in *like a*. If the word is not in ConceptNet, we fall back to its synonyms obtained from WordNet (Miller, 1995).

3. **Metaphor Masking (META_M)**: The third baseline is the metaphor generation model from a literal sentence described by Stowe et al. (2020). Following their approach, we fine-tune BART where we mask the adjective or adverb in the end of the literal sentence. The input is the masked text, with the hidden adjective or adverb (*The city was* <***MASK*** >), and the output is the original simile (*The city was like a painting*). Through this learning paradigm, the model learns that it needs to generate simile when it encounters the mask token. At test time, we provide the model with the literal input, mask the adjective/adverb, and the model produces an output conditioned on the adjective/adverb masking training.

[10]ConceptNet is a weighted graph with multiple relations as can be viewed here http://conceptnet.io/. We use 'has property" for our work. There are multiple edges for objects with their properties. We choose the edge with the highest weight

	B-1	B-2	BERT-S	NOVELTY
RTRVL	0.0	0.0	0.13	92.6
BART	3.25	0.32	0.12	92.6
META_M	3.73	0.96	0.15	**93.3**
SCOPE	**8.03**	**3.59**	**0.18**	88.6

Table 2: Results using automatic metrics: BLEU-1 (B-1), BLEU-2 (B-2), BERTScores (BERT-S) and Novelty. Boldface denotes the best results.

3.2 Evaluation Criteria

Automatic evaluation. *BLEU* (Papineni et al., 2002) is one of the most widely used automatic evaluation metric for generation tasks such as Machine Translation. However, for creative text generation, it is not ideal to expect significant n-gram overlaps between the machine-generated and the gold-standard sentences. We still report the BLEU scores for generated VEHICLE after discarding the common prefix with the gold.

BERTScore (Zhang et al., 2019) has been used recently for evaluating text generation using contextualized embeddings and it is said to somewhat ameliorate the problems with BLEU. It computes a similarity score using contextual embeddings for each token in the candidate (here VEHICLE in the generated simile) with each token in the reference (VEHICLE in the human written simile). To compute F1-Score it uses Recall (matching each token in reference to a token in candidate) and Precision (matching each token in candidate to a token in reference). We report F1-Score of *BERTScore.*

Novelty. To measure the model's generalization capability, we also want to test how well our models can generate novel content. We capture the proportion of generated VEHICLE conditioned on an adverb/adjective literal PROPERTY that does not appears in the training set.

Human evaluation. Automated metrics are not adequate on their own for evaluating methods to generate creative text so we present a human-based evaluation as well. We evaluate on a total of 900

System	C	R1	R2	OQ
HUMAN1	**3.61** (0.34)	**3.74** (0.43)	3.90 (0.51)	**3.54** (0.40)
HUMAN2	3.46 (0.31)	3.72 (0.43)	**3.97** (0.47)	3.44 (0.39)
RTRVL	1.90 (0.39)	1.85 (0.44)	1.73 (0.50)	1.85 (0.42)
BART	2.68 (0.39)	2.78 (0.45)	2.75 (0.51)	2.61 (0.41)
META_M	2.68 (0.42)	2.72 (0.46)	2.77 (0.47)	2.59 (0.41)
SCOPE	3.16 (0.35)	3.50 (0.43)	3.78 (0.52)	3.32 (0.43)

Table 3: Human evaluation on several criteria of similes' quality for different systems' outputs and human written similes. We show average scores on a 1-5 scale with 1 denotes the worst and 5 be the best; the corresponding inter-annotator agreement (IAA) is in the parenthesis. Boldface denotes the best results and underscore denotes the second bests.

	SCOPE/H1		SCOPE/H2		SCOPE/META_M	
	w%	l%	w%	l%	w%	l%
C	28.0	**58.6**	26.6	**57.3**	**58.6**	31.3
R1	37.3	**51.3**	33.3	**50.0**	**63.3**	18.0
R2	42.6	**45.3**	37.3	**44.6**	**69.3**	17.3
OQ	32.6	**54.6**	41.3	**50.0**	**68.6**	18.6

Table 4: Pairwise comparison between SCOPE and HUMAN1(H1), HUMAN2(H2), and META_M. Win[w]% (lose[l]%) is the percentage of SCOPE gets a higher (lower) average score compared to HUMAN1, HUMAN2 and META_M. The rest are ties.

utterances, 600 generated from 4 systems and 300 utterances generated by humans. We proposed a set of 4 criteria to evaluate the generated output: (1) **Creativity (C)** ("How creative are the utterances?"), (2) **Overall Quality (OQ)** ("How good is the simile overall? (*MTurk guidelines were to score based on how creative, well formed, meaningful and relevant it is with respect to the literal utterance*)), (3) **Relevance1 (R1)** ("How relevant is the generated VEHICLE in terms of portraying the PROPERTY?") and (4) **Relevance2 (R2)** ("How relevant is the VEHICLE to the TOPIC in the generation?"). As we evaluate on 4 separate dimensions for 900 utterances we have a total of 3600 evaluations. We hired Turkers on MTurk to rate outputs from the 4 systems and 2 humans. Each Turker was given the literal utterance as well as the 6 generated similes (randomly shuffled) Each criteria was rated on a scale from 1 (not at all) to 5 (very). Each utterance was rated by three separate Turkers. We hired 86, 48, 42, 46 Turkers for the tasks of Creativity, Overall Quality, Relevance1, Relevance2 respectively. Further details in Appendix A.4 .

4 Experimental Results

4.1 Automatic Evaluation

Table 2 shows BLEU-1, BLEU-2 and BERTScore of our system compared to the three baselines. The low scores can be attributed to the nature of creative NLG tasks. To further validate this, we also compute the BLEU-1 and BLEU-2 score between the two literary experts treating one as reference and other as candidate and get scores of 4.12 and 0.52 respectively. BERTScore is often a better metric as it utilizes contextualized embeddings. For example for a candidate [**desert**] with multi-reference as [[**sandy death trap**],[**wasteland**]] , we get a BERTscore of 0.99 while BLEU score is 0.0. Finally our best model SCOPE emerges as the winner for both BLEU and BERTScore. For novelty, SCOPE can still generate novel content 88% of the time proving it is generalizable to unseen test data. Furthermore, there are 5,558 unique PROPERTY in training data and 41% of PROPERTY in testing data does not appear in training, showing our model is generalizable to unseen PROPERTY as well.

4.2 Human Evaluation Scores

Table 3 presents the scores of the aforementioned human evaluation criteria for our model and the baselines on the test set. The results show that SCOPE is significantly ($p < .001$ according to approximate randomization test) better than the baselines on all four criteria. For all metrics our best system is comparable to humans. We also computed Pearson's correlation between OQ with other metrics and observed that R1 and R2 had moderate correlation of 0.54 and 0.52 with OQ, while C was fairly correlated (0.31) to OQ suggesting a relevance matters when deciding the quality of a simile.

Pairwise Comparison between systems. Table 4 shows the pairwise comparisons between SCOPE's output and human generated similes (HUMAN1 and HUMAN2), and META_M (Stowe et al., 2020), respectively. Given a pair of inputs, we decide win/lose/tie by comparing the average scores (over three Turkers) of both outputs. We see that SCOPE outperforms META_M on all the metrics. For overall quality, although it is a given that literary experts are better, our SCOPE model still has a winning rate of 32.6% and 41.3%, respectively against the two humans.

Literal	System	Simile	R1	R2	C	OQ
It was obscene, but she was drawn to it, *fascinated*	HUMAN1	It was obscene, but she was drawn to it like a *moth to a flame*	**5.0**	**4.0**	3.0	**4.7**
	HUMAN2	It was obscene, but she was drawn to it like it was *a bad boy in leather jacket*	4.0	3.3	**4.3**	1.7
	RTRVL	It was obscene, but she was drawn to it like a *read*	1.0	1.0	1.3	1.3
	BART	It was obscene, but she was drawn to it like a *magnet*	**5.0**	**4.0**	2.7	2.7
	META_M	It was obscene, but she was drawn to it like a *magnet*	**5.0**	**4.0**	2.7	2.7
	SCOPE	It was obscene, but she was drawn to it like a *moth to a flame*	**5.0**	**4.0**	3.0	**4.7**
I start to prowl across the room *warily*	HUMAN1	I start to prowl across the room like a *tightrope walker on dental floss*	3.7	**4.0**	2.7	**5.0**
	HUMAN2	I start to prowl across the room like a *nervous criminal*	**4.7**	3.7	2.7	4.0
	RTRVL	I start to prowl across the room like a ——	2.0	1.0	2.7	1.0
	BART	I start to prowl across the room like a *cat*	2.7	3.3	3.7	3.3
	META_M	I start to prowl across the room like a *lion*	2.7	3.7	3.3	2.7
	SCOPE	I start to prowl across the room like a *cat stalking its prey*	3.0	**4.0**	**4.0**	4.0
If it falls into the wrong hands it would be *catastrophic*	HUMAN1	If it falls into the wrong hands it would be like a *nuclear apocalyse*	4.0	4.3	**4.7**	**5.0**
	HUMAN2	If it falls into the wrong hands it would be like *World War III*	**4.3**	**4.7**	4.0	4.7
	RTRVL	If it falls into the wrong hands it would be like a *police officer*	1.3	1.0	1.3	1.0
	BART	If it falls into the wrong hands it would be like a *gift to 'terrorists'*	3.7	4.0	2.3	4.0
	META_M	If it falls into the wrong hands it would be like a *gift*	1.3	1.3	1.7	1.0
	SCOPE	If it falls into the wrong hands it would be like a *nuclear bomb*	**4.3**	**4.7**	4.0	**5.0**
Having a thin figure, he looked *unpleasant*	HUMAN1	Having a thin figure, he looked like a *dry, overgrown blade of grass*	2.3	3.7	**4.7**	**4.3**
	HUMAN2	Having a thin figure, he looked like a *couch without cushions*	2.0	**4.7**	4.0	3.0
	RTRVL	Having a thin figure, he was looked like a *pain*	2.3	1.0	2.0	1.3
	BART	Having a thin figure, he looked like a *man*	2.3	1.0	1.0	1.7
	META_M	Having a thin figure, he looked like a *child*	2.0	2.3	1.3	2.7
	SCOPE	Having a thin figure, he looked like a *walking corpse*	**4.0**	4.3	3.0	3.7
Keaton was lying face down on the ground *lamely*	HUMAN1	Keaton was lying face down on the ground like a *snapped twig*	1.3	4.0	3.7	2.3
	HUMAN2	Keaton was lying face down on the ground like a *maimed animal*	3.7	**4.3**	**4.0**	3.3
	RTRVL	Keaton was lying face down on the ground like a *life*	1.3	1.0	2.3	1.0
	BART	Keaton was lying face down on the ground like a *zombie*	3.3	2.7	3.0	3.0
	META_M	Keaton was lying face down on the ground like a *sack of potatoes*	**4.0**	3.7	2.3	3.3
	SCOPE	Keaton was lying face down on the ground like a *limp corpse*	**4.0**	4.0	3.7	**4.3**
As a human tick hybrid, life can sometimes be *tricky*	HUMAN1	As a human tick hybrid, life can sometimes be like *sucking the nectar out of dried fruit*	2.3	3.7	**4.7**	3.3
	HUMAN2	As a human tick hybrid, life can sometimes be like *interspecies balancing act*	**4.3**	4.3	3.0	4.0
	RTRVL	As a human tick hybrid, life can sometimes be like a *ceiling*	1.3	1.0	1.3	1.7
	BART	As a human tick hybrid, life can sometimes be like a *zoo*	2.3	2.7	2.3	2.7
	META_M	As a human tick hybrid, life can sometimes be like *dream*	1.3	2.3	2.0	2.0
	SCOPE	As a human tick hybrid, life can sometimes be like a *slippery slope*	**4.3**	**4.7**	2.7	**4.3**

Table 5: Examples of generated outputs from different systems (with human written similes as references). We show average scores (over three annotators) on a 1-5 scale with 1 denotes the worst and 5 be the best. The italics texts in the literal column represent the PROPERTY while those in Simile column represents the generated VEHICLE. Boldface indicates the best results. More examples in Appendix A.3

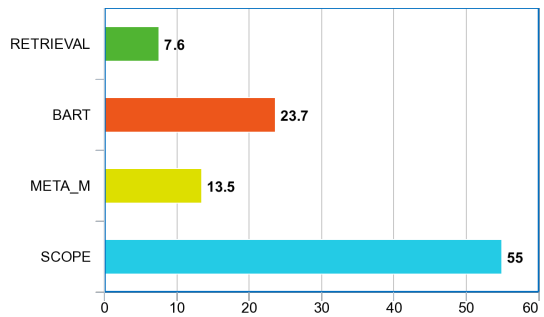

Figure 4: Barchart showing the percent of times each individual system won in terms of Overall Quality.

Storyline: sky → sunset → walk → walked → beautiful
The sky was ~~beautiful~~ [*like a blue canvas*]. Jane wanted to see the sunset. She decided to go for a walk.She walked for a long time.When she was done she saw the sunset was beautiful.
Title: car accident **Storyline**: driving → hit → hit → car → fixed
Tom was driving down the road. Suddenly he hit a tree. He swerved and hit a pole. Tom's car was ~~totaled~~ *[like a wreck]*. Luckily he was able to get it fixed

Table 6: An example of a GPT-2 generated short stories with the title **Sunset** and **Car Accident**. We replace the literal sentences with generated similes from SCOPE.

5 Qualitative Analysis

Table 9 demonstrates several generation outputs from different systems along with human judgements on individual criteria. We observe that often our model is better than at least one human on a certain criteria while outperforming the baselines by a large margin.

5.1 Role of Relevance

While conditioning on the context of literal sentences might lead to grammatically correct similes, they are often not meaningful and relevant to the PROPERTY in consideration. META_M generates similes by fine-tuning BART on literal sentences where the common sense PROPERTY is masked. The lack of relevance mapping during fine-tuning often leads to improper generations. For instance, referring to Table 9, the context of 'falling into the wrong hands' is more likely to lead to something bad and hence here 'gift' (generated by META_M) is not appropriate while 'nuclear bomb' (generated by our model) is. One explanation is that our SCOPE model uses common sense knowledge as a way of incorporating relevance.

5.2 Role of Context

The role of context is necessary for simile generation. For example given the literal input *'But times are hard, and silver bullets are **expensive**'* even though ConceptNet tells us **diamonds** are objects with *HasProperty* expensive, a generated simile by RTRVL model *'But times are hard, and silver bullets are like a **diamond**'* seems inappropriate suggesting that a context leads to better generation. Our SCOPE model generates *'But times are hard, and silver bullets are like a **luxury item**'*

6 Task-based Evaluation: Simile for Story Generation

Similes are often used to evoke imagery. Generating or transforming text to be evocative can be useful for computational journalism (Spangher et al.), poetry generation (Ghazvininejad et al., 2017; Van de Cruys, 2020) and story writing (Peng et al., 2018; Yao et al., 2019; Goldfarb-Tarrant et al., 2020). Table 10 shows how we can use our simile generation module as a post processing step to replace literal sentences leading to more expressive and creative stories. To further test this hypothesis we conduct an experiment further outlined below.

GPT2	GPT2+META_M	GPT2+SCOPE
23%	25%	**42%**

Table 7: Win% (in terms of average score over three annotators) of stories generated with only GPT2, GPT2 with META_M or SCOPE simile post processing. The rest are ties.

6.1 Story Generation

We use the ROCStories (Mostafazadeh et al., 2016) dataset to generate stories using the *Plan and Write* model outlined by Yao et al. (2019); Goldfarb-Tarrant et al. (2019). We introduce a two step pipeline procedure where we fine-tune a pre-trained GPT2 (Radford et al., 2018) model on titles and storyline from the training set to generate a storyline given a title (Row 1 Table 10). In parallel, we also fine-tune GPT2 on storylines and stories from the training set to generate a story given a storyline (Row 2 Table 10). At test time, we generate a storyline using an input title first and then use the generated storyline to generate a story.

6.2 Post Processing

There can be multiple sentences ending with an adjective or adverb and replacing each of them with a simile might lead to over-embellishment. Under such situations we feed only one randomly selected sentence to SCOPE and META_M module and replace the sentence in GPT2 generated story with the output from SCOPE or META_M, respectively.

6.3 Human evaluation.

We randomly select 50 titles from ROCStories data set and generate stories as described above. We postprocess it using both SCOPE and META_M separately. Thus for each title we have 3 stories 1) the original GPT2 story, 2) the GPT2 story postprocessed with SCOPE, and 3) the GPT2 story postprocessed with META_M. For each given titles, we present these 3 stories to workers in AMT and ask them to score them in a range of 1(poor) to 5 (excellent) based on creativity and evocativeness. Experimental results from Table 7 prove that effective usage of similes can improve evocativeness of machine generated stories.

7 Related Work

Simile generation is a relatively new task. Most prior work has focused on detection of similes. The closest task in NLP to simile generation is generating metaphors. However, it should be noted that

the overlap between the expressive range of similes and metaphors is known to be only partial: there are similes that cannot be rephrased as metaphors, similarly the other way around (Israel et al., 2004).

7.1 Simile Detection and Analysis

Niculae and Danescu-Niculescu-Mizil (2014) proposed frameworks for annotating similes from product reviews by considering their semantic and syntactic characteristics as well as the challenges inherent to the automatic detection of similes. Qadir et al. (2015, 2016) built computational models to recognize affective polarity and implicit properties in similes. Unlike these works, we focus on generating similes by transforming a literal sentence while still being faithful to the property in context.

7.2 Metaphor Generation

Earlier works in metaphor generation (Abe et al., 2006; Terai and Nakagawa, 2010) were conducted on a lexical or phrase level, using template and heuristic-based methods. (Gero and Chilton, 2019) presented an interactive system for collaboratively writing metaphors with a computer. They use an open source knowledge graph and a modified Word Mover's Distance algorithm to find a large, ranked list of suggested metaphorical connections. Word embedding approaches (Gagliano et al., 2016) have also been used for metaphor generation. (Young, 1987) also present a relational data base method for automatic metaphor generation. However, the metaphors generated through these methods do not take semantic context into consideration and lack the flexibility and creativity necessary to instantiate similes through a natural language sentence.

Yu and Wan (2019) use neural models to generate metaphoric expressions given a literal input in an unsupervised manner. Stowe et al. (2020) develop a new framework dubbed 'metaphor masking' where they train a supervised seq2seq model with input as the masked text, where they mask or hide the metaphorical verb while preserving the original text as the output. However, both these works hinge on metaphoric verbs unlike similes where we not only need to replace the literal property with a vehicle but it also needs to be relevant to the context and the tenor. Additionally, we also use (Stowe et al., 2020) as a baseline and show that our approach leads to better similes using both quantitative and qualitative evaluation metrics.

8 Conclusion

We establish a new task for NLG: simile generation from literal sentences. We propose a novel way of creating parallel corpora and a transfer-learning approach for generating similes. Human and automatic evaluations show that our best model is successful at generating similes. Our experimental results further show that to truly be able to generate similes based on actual metaphoric or conceptual mappings, it is important to incorporate some common sense knowledge about the topics and their properties. Future directions include exploration of other knowledge bases to help the inference process and applying our simile generation approach to different creative NLG tasks such as pun (He et al., 2019), sarcasm (Chakrabarty et al., 2020), and hyperbole (Troiano et al., 2018).

Acknowledgments

This work was supported by the CwC program under Contract W911NF-15-1-0543 with the US Defense Advanced Research Projects Agency (DARPA). The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government. The authors would like to thank Kai-Wei Chang, Christopher Hidey, Christopher Robert Kedzie, Anusha Balakrishnan and Liunian Harold Li for useful discussions. The authors also thank the members of PLUSLab at the University of California Los Angeles and University of Southern California and the anonymous reviewers for helpful comments. The first author also wants to acknowledge his father Tridib Chakrabarty who bravely fought through this pandemic and left us for the heavenly abode through the period of writing this paper.

References

Keiga Abe, Kayo Sakamoto, and Masanori Nakagawa. 2006. A computational model of the metaphor generation process. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 28.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Tuhin Chakrabarty, Debanjan Ghosh, Smaranda Muresan, and Nanyun Peng. 2020. R^3: Reverse, retrieve, and rank for sarcasm generation with commonsense knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7976–7986, Online. Association for Computational Linguistics.

Tim Van de Cruys. 2020. Automatic poetry generation from prosaic text. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2471–2480.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2017. Style transfer in text: Exploration and evaluation. In *Proceedings of AAAI*.

Andrea Gagliano, Emily Paul, Kyle Booten, and Marti A Hearst. 2016. Intersecting word vectors to take figurative language to new heights. In *Proceedings of the Fifth Workshop on Computational Linguistics for Literature*, pages 20–31.

Katy Ilonka Gero and Lydia B. Chilton. 2019. Metaphoria: An algorithmic companion for metaphor creation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*, page 296. ACM.

Marjan Ghazvininejad, Xing Shi, Jay Priyadarshi, and Kevin Knight. 2017. Hafez: an interactive poetry generation system. In *Proceedings of ACL 2017, System Demonstrations*, pages 43–48, Vancouver, Canada. Association for Computational Linguistics.

Seraphina Goldfarb-Tarrant, Tuhin Chakrabarty, Ralph Weischedel, and Nanyun Peng. 2020. Content planning for neural story generation with aristotelian rescoring. In *the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Seraphina Goldfarb-Tarrant, Haining Feng, and Nanyun Peng. 2019. Plan, write, and revise: an interactive system for open-domain story generation. In *2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2019), Demonstrations Track*, volume 4, pages 89–97.

Patrick Hanks. 2013. *Lexical analysis: Norms and exploitations*. Mit Press.

He He, Nanyun Peng, and Percy Liang. 2019. Pun generation with surprise. In *2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2019)*, volume 1.

Michael Israel, Jennifer Riddle Harding, and Vera Tobin. 2004. On simile.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: A simple approach to sentiment and style transfer. *arXiv preprint arXiv:1804.06437*.

Lizhen Liu, Xiao Hu, Wei Song, Ruiji Fu, Ting Liu, and Guoping Hu. 2018. Neural multitask learning for simile recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1543–1553, Brussels, Belgium. Association for Computational Linguistics.

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

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and evaluation framework for deeper understanding of commonsense stories. *arXiv preprint arXiv:1604.01696*.

Suzanne Mpouli. 2017. Annotating similes in literary texts. In *Proceedings of the 13th Joint ISO-ACL Workshop on Interoperable Semantic Annotation (ISA-13)*.

Vlad Niculae and Cristian Danescu-Niculescu-Mizil. 2014. Brighter than gold: Figurative language in user generated comparisons. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2008–2018, Doha, Qatar. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*.

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. Association for Computational Linguistics.

Anthony M Paul, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 1970. Figurative language. In *Philosophy & Rhetoric*, pages 225–248.

Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. 2018. Towards controllable story generation. In *NAACL Workshop*.

Ashequl Qadir, Ellen Riloff, and Marilyn A Walker. 2015. *Learning to recognize affective polarity in similes*.

Ashequl Qadir, Ellen Riloff, and Marilyn A Walker. 2016. *Automatically inferring implicit properties in similes*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in neural information processing systems*, pages 6830–6841.

Alexander Spangher, Nanyun Peng, and Emilio Ferrara. Modeling "newsworthiness" for lead-generation across corpora.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. *In Thirty-First AAAI Conference on Artificial Intelligence.*

Kevin Stowe, Leonardo Ribeiro, and Iryna Gurevych. 2020. Metaphoric paraphrase generation. *arXiv preprint arXiv:2002.12854*.

Akhilesh Sudhakar, Bhargav Upadhyay, and Arjun Maheswaran. 2019. Transforming delete, retrieve, generate approach for controlled text style transfer. *arXiv preprint arXiv:1908.09368*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Asuka Terai and Masanori Nakagawa. 2010. A computational system of metaphor generation with evaluation mechanism. In *International Conference on Artificial Neural Networks*, pages 142–147. Springer.

Enrica Troiano, Carlo Strapparava, Gözde Özbal, and Serra Sinem Tekiroğlu. 2018. A computational exploration of exaggeration. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3296–3304, Brussels, Belgium. Association for Computational Linguistics.

Tony Veale. 2013. Humorous similes. *Humor*, 26(1):3–22.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7378–7385.

Lawrence F Young. 1987. The metaphor machine: A database method for creativity support. *Decision Support Systems*, 3(4):309–317.

Zhiwei Yu and Xiaojun Wan. 2019. How to avoid sentences spelling boring? towards a neural approach to unsupervised metaphor generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 861–871, Minneapolis, Minnesota. Association for Computational Linguistics.

Jiali Zeng, Linfeng Song, Jinsong Su, Jun Xie, Wei Song, and Jiebo Luo. 2019. Neural simile recognition with cyclic multitask learning and local attention. *arXiv preprint arXiv:1912.09084*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. *arXiv preprint arXiv:1903.00138*.

A Appendix

A.1 Hyper-Parameters and Other Experimental Settings

For retrieving commonsense properties of the vehicle, we use the pre-trained COMET model [11] and retrieve top 5 candidates for each input.

1. **Number of Parameters:** For BART we use the BART large checkpoint (400M parameters) and use the implementation by FAIRSEQ (Ott et al., 2019).[12]

2. **Number of Epochs:** We fine-tune pre-trained BART for 17 epochs for SCOPE model.

3. **Training Time:** Our training time is 52 minutes.

4. **Hardware Configuration:** We use 4 RTX 2080 GPU,

5. **Training Hyper parameters:** We use the same parameters mentioned in the github repo where BART was fine-tuned for CNN-DM summarization task with the exception of MAX-TOKENS (size of each mini-batch, in terms of the number of tokens) being 1024 for us.

6. **Decoding Strategy & Hyper Parameters:** For decoding we generate similes from our models using a top-k random sampling

[11] https://github.com/atcbosselut/comet-commonsense

[12] https://github.com/pytorch/fairseq/tree/master/examples/bart

	#WORKERS	α
C	86	0.36
OP	48	0.41
R1	42	0.44
R2	46	0.49

Table 8: C, R1,R2 and OQ denote Creativity, Relevance of Vehicle w.r.t Property, Relevance of Tenor to Vehicle and Overall Quality. WORKERS denote number of workers employed for each task and α denotes Krippendorff's alpha (α) , reliability coefficient used for our study

scheme (Fan et al., 2018). At each timestep, the model generates the probability of each word in the vocabulary being the likely next word. We randomly sample from the k = 5 most likely candidates from this distribution. We also use a softmax temperature of 0.7.

A.2 Dataset Assumptions

Figure 5: Proprty associated with fool

While distant supervision is often used to collect a lot of data without human/ expert annotation through this process we introduce, noise in our self labeled similes. For example the sentence *I feel like a fool* is ideally not a simile.We notice 1.1% of the training data with PNP in `TOPIC` and typically $<= 6$ in token count such as *I would like a , I don't like a, I feel like a, I think like a*. However our transformation method still works here. Based on Figure 5 we see the common sense properties associated for *fool* are *sneaky, stupid,funny,dangerous, bad*. Our best literal transformation for *I feel like a fool* is then *I feel stupid*. So even though there is some noise this method still benefits our training procedure

A.3 Examples

Table 9 shows generations from all 4 systems along with gold similes and how turkers scored them on a scale of 1 to 5 for C,R1,R2 and OQ.

A.4 Amazon Mechanical Turk Settings

The 2nd column Table 8 shows the number of distinct workers employed for each task. Column 3 shows inter-rater agreement between workers. Except for Creativity, for the other 3 tasks workers are moderately correlated. For creativity workers are fairly correlated.

Figure 6,7,8 and 9 show the Amazon Mechanical Turk interfaces for the tasks of *Creativity (C)* ("How creative are the utterances ?"), (2) *Relevance1 (R1)* ("How relevant is the generated `VEHICLE` in terms of portraying the `PROPERTY`?") and (3) *Relevance2 (R2)* ("How relevant is the `VEHICLE` to the `TOPIC` in the generation?") (4) *Overall Quality (OQ)* ("How good is the simile overall ?". As can be seen we provide with explicit examples and a clear description of the task to turkers. We also mention and highlight the importance of evaluating similes along with input and not in isolation.

A.5 GPT2 generated stories preprocessed with SCOPE

Table 10 shows several example stories where a literal sentence has been replaced by a simile.

Literal	System	Simile	R1	R2	C	OQ
From the day you were born, you've been *invincible*	HUMAN1	From the day you were born, you've been like a *well-seasoned superhero*	4.0	4.0	4.0	3.3
	HUMAN2	From the day you were born, you've been like *Superman*	**4.7**	4.7	**4.3**	1.7
	RTRVL	From the day you were born, you've been like a ——-	1.0	4.0	1.0	1.0
	BART	From the day you were born, you've been like a *son*	**5.0**	1.3	1.0	1.3
	META_M	From the day you were born, you've been like a *son to me*	1.0	1.0	1.7	1.3
	SCOPE	From the day you were born, you've been like a *superhero*	**4.7**	**5.0**	4.0	**4.3**
For centuries, the Tyrant has made life *miserable*	HUMAN1	For centuries, the Tyrant has made life like an *impatient storm reaching to be a hurricane*	2.7	2.7	4.0	3.3
	HUMAN2	For centuries, the Tyrant has made life like a *dreary prison*	**5.0**	3.7	**4.3**	3.7
	RTRVL	For centuries, the Tyrant has made life like a ——	1.3	1.0	1.3	1.7
	BART	For centuries, the Tyrant has made life like a *prison*	4.3	3.0	3.3	3.3
	META_M	For centuries, the Tyrant has made life like a *prison in this country*	4.3	3.0	3.3	3.7
	SCOPE	For centuries, the Tyrant has made life like a *living hell*	4.7	**5.0**	4.0	**4.0**
Adrenaline shot through him *powerful*	HUMAN1	Adrenaline shot through him like a *lightning bolt*	3.7	**4.3**	3.3	**4.7**
	HUMAN2	Adrenaline shot through him like a *hypodermic injection*	3.7	3.7	3.0	2.7
	RTRVL	Adrenaline shot through him like a *natural energy*	2.7	2.3	2.7	2.7
	BART	Adrenaline shot through him like a *bullet*	**4.3**	**4.3**	**4.3**	3.7
	META_M	Adrenaline shot through him like a *bullet*	**4.3**	**4.3**	**4.3**	3.7
	SCOPE	Adrenaline shot through him like a *bolt of lightning*	3.3	**4.3**	4.0	**4.7**
Constructing the flat pack TV cabinet was meant to be *easy*	HUMAN1	Constructing the flat pack TV cabinet was meant to be like *putting on velcro shoes*	4.0	4.7	**5.0**	3.7
	HUMAN2	Constructing the flat pack TV cabinet was meant to be like *turning on a light*	**4.3**	4.0	3.0	3.0
	RTRVL	Constructing the flat pack TV cabinet was meant to be like a *learn to change car tire*	2.0	2.3	2.3	2.7
	BART	Constructing the flat pack TV cabinet was meant to be like a *Lego set*	3.0	2.3	2.7	4.7
	META_M	Constructing the flat pack TV cabinet was meant to be like a *house*	1.0	1.0	1.3	2.3
	SCOPE	Constructing the flat pack TV cabinet was meant to be like a *cake walk*	**5.0**	**4.3**	3.0	**4.3**
You are an oracle whose predictions have always come *true*	HUMAN1	You are an oracle whose predictions have always come true like the *rising sun*	3.7	**4.3**	**4.0**	2.7
	HUMAN2	You are an oracle whose predictions have always come true like *highly researched hypotheses*	3.0	4.0	1.7	2.7
	RTRVL	You are an oracle whose predictions have always come true like a *fact*	3.7	**4.3**	2.0	3.0
	BART	You are an oracle whose predictions have always come true like a *man of action*	2.0	2.0	2.7	2.0
	META_M	You are an oracle whose predictions have always come true like a *bolt from the blue*	2.7	2.3	3.3	3.0
	SCOPE	You are an oracle whose predictions have always come true like a *prophecy*	3.0	2.7	3.7	**4.0**

Table 9: Examples of generated outputs from different systems (with human written similes as references).We show average scores (over three annotators) on a 1-5 scale where 1 denotes the worst and 5 be the best. The italics texts in the literal column represent the PROPERTY while those in Simile column represents the generated VEHICLE. Boldface indicates the best results.

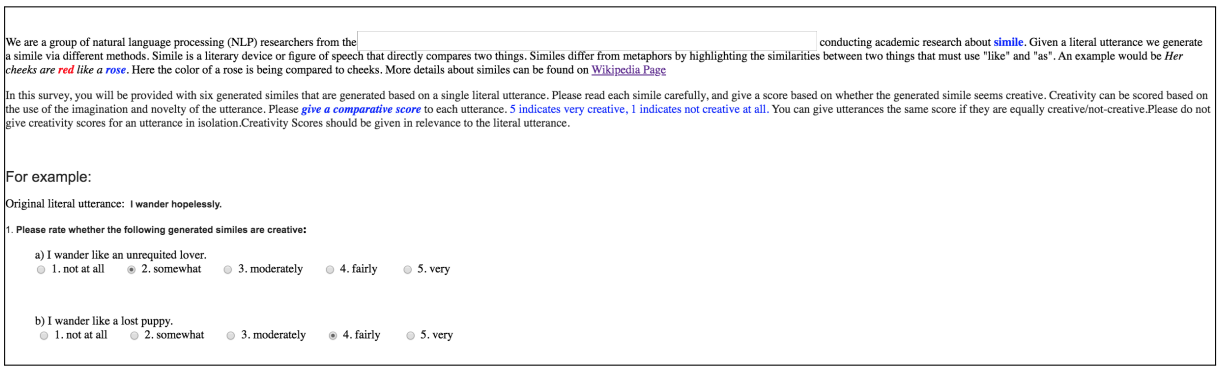

Figure 6: MTurk interface for scoring **Creativity**

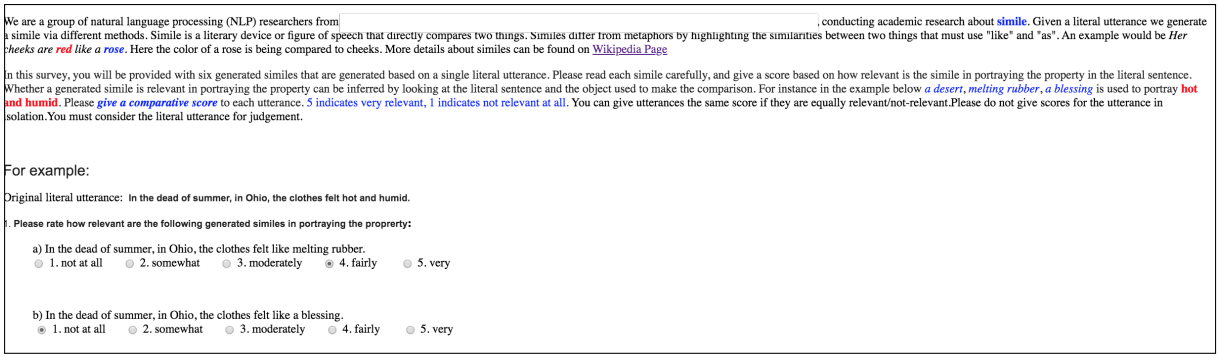

Figure 7: MTurk interface for scoring **Relevance1**

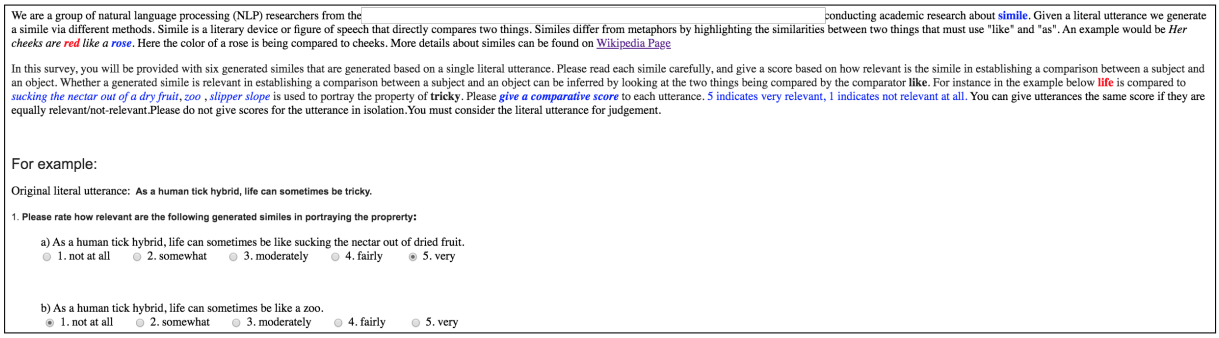

Figure 8: MTurk interface for scoring **Relevance2**

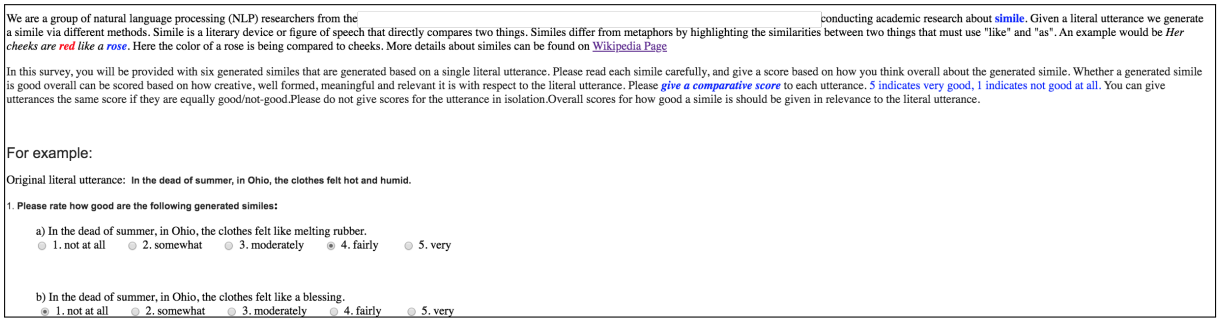

Figure 9: MTurk interface for scoring **Overall Quality**

Title: a gift from the mentor **Storyline**: loved → playing → promised → tried → surprised
Harry loved playing tennis. One day while playing he broke his racket. His coach had promised to buy him a new racket if he practiced. Harry tried hard to practice and was confident in his new racket. To his surprise his coach bought him a racket for his birthday and he was ~~ecstatic~~ *[like a child on Christmas day]*
Title: The pet bug **Storyline**: playing → caught → bug → hoped → release
Oliver was playing in his yard. Suddenly he spotted a bug he hadn't caught. The bug was a big beetle. He hoped it would be there ~~forever~~ *[like a shadow]*. But unfortunately it was too late to release it
Title: fishing **Storyline**: fish → lake → kids → caught → home
The kids were great at catching fish. They woke up early and packed up their tackle box and hiked to the lake. The kids set up their lures and caught as many as they could. The fish were all caught and the kids laughed ~~heartily~~ *[like a group of hyenas]*. They went home and had a great day fishing

Table 10: Example of a GPT-2 generated short story on respective title , storyline. We replace the first literal sentence with a generated simile from SCOPE.