Delete, Retrieve, Generate: A Simple Approach to Sentiment and Style Transfer

Juncen Li[*1] **Robin Jia**[2] **He He**[2] **Percy Liang**[2]
[1] WeChat Search Application Department, Tencent
[2] Computer Science Department, Stanford University
`juncenli@tencent.com`
`{robinjia,hehe,pliang}@cs.stanford.edu`

Abstract

We consider the task of text attribute transfer: transforming a sentence to alter a specific attribute (e.g., sentiment) while preserving its attribute-independent content (e.g., changing *"screen is just the right size"* to *"screen is too small"*). Our training data includes only sentences labeled with their attribute (e.g., positive or negative), but not pairs of sentences that differ only in their attributes, so we must learn to disentangle attributes from attribute-independent content in an unsupervised way. Previous work using adversarial methods has struggled to produce high-quality outputs. In this paper, we propose simpler methods motivated by the observation that text attributes are often marked by distinctive phrases (e.g., *"too small"*). Our strongest method extracts content words by deleting phrases associated with the sentence's original attribute value, retrieves new phrases associated with the target attribute, and uses a neural model to fluently combine these into a final output. On human evaluation, our best method generates grammatical and appropriate responses on 22% more inputs than the best previous system, averaged over three attribute transfer datasets: altering sentiment of reviews on Yelp, altering sentiment of reviews on Amazon, and altering image captions to be more romantic or humorous.

1 Introduction

The success of natural language generation (NLG) systems depends on their ability to carefully control not only the topic of produced utterances, but also attributes such as sentiment and style. The desire for more sophisticated, controllable NLG has led to increased interest in text attribute transfer—the task of editing a sentence to alter specific attributes, such as style, sentiment, and tense (Hu et al., 2017; Shen et al., 2017; Fu et al., 2018). In each of these cases, the goal is to convert a sentence with one attribute (e.g., negative sentiment) to one with a different attribute (e.g., positive sentiment), while preserving all attribute-independent content[1] (e.g., what properties of a restaurant are being discussed). Typically, aligned sentences with the same content but different attributes are not available; systems must learn to disentangle attributes and content given only unaligned sentences labeled with attributes.

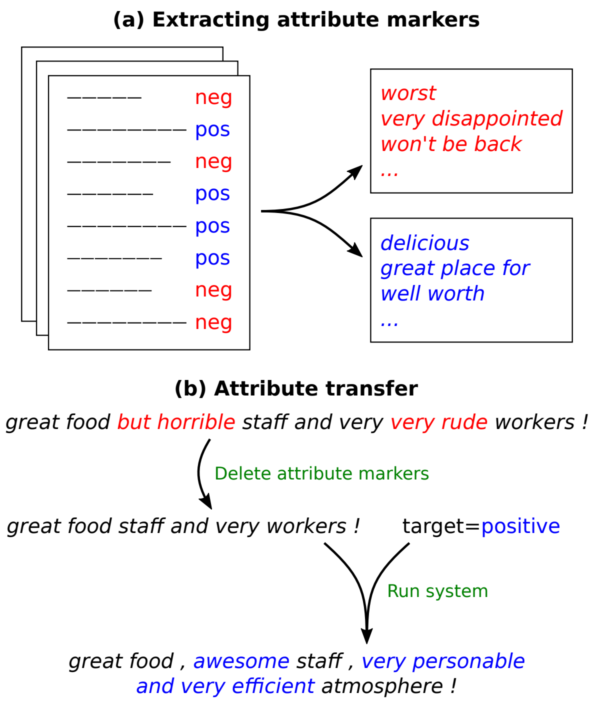

(a) Extracting attribute markers
neg
pos
neg
pos
pos
pos
neg
neg
worst
very disappointed
won't be back
...
delicious
great place for
well worth
...
(b) Attribute transfer
great food but horrible staff and very very rude workers !
Delete attribute markers
great food staff and very workers !
target=positive
Run system
great food , awesome staff , very personable and very efficient atmosphere !

Figure 1: An overview of our approach. (a) We identify attribute markers from an unaligned corpus. (b) We transfer attributes by removing markers of the original attribute, then generating a new sentence conditioned on the remaining words and the target attribute.

Previous work has attempted to use adversarial

[*]Work done while the author was a visiting researcher at Stanford University.

[1] Henceforth, we refer to attribute-independent content as simply *content*, for simplicity.

networks (Shen et al., 2017; Fu et al., 2018) for this task, but—as we demonstrate—their outputs tend to be low-quality, as judged by human raters. These models are also difficult to train (Salimans et al., 2016; Arjovsky and Bottou, 2017; Bousmalis et al., 2017).

In this work, we propose a set of simpler, easier-to-train systems that leverage an important observation: attribute transfer can often be accomplished by changing a few *attribute markers*—words or phrases in the sentence that are indicative of a particular attribute—while leaving the rest of the sentence largely unchanged. Figure 1 shows an example in which the sentence of a sentence can be altered by changing a few sentiment-specific phrases but keeping other words fixed.

With this intuition, we first propose a simple baseline that already outperforms prior adversarial approaches. Consider a sentiment transfer (negative to positive) task. First, from unaligned corpora of positive and negative sentences, we identify attribute markers by finding phrases that occur much more often within sentences of one attribute than the other (e.g., *"worst"* and *"very disppointed"* are negative markers). Second, given a sentence, we delete any negative markers in it, and regard the remaining words as its content. Third, we retrieve a sentence with similar content from the positive corpus.

We further improve upon this baseline by incorporating a neural generative model, as shown in Figure 1. Our neural system extracts content words in the same way as our baseline, then generates the final output with an RNN decoder that conditions on the extracted content and the target attribute. This approach has significant benefits at training time, compared to adversarial networks: having already separated content and attribute, we simply train our neural model to reconstruct sentences in the training data as an auto-encoder.

We test our methods on three text attribute transfer datasets: altering sentiment of Yelp reviews, altering sentiment of Amazon reviews, and altering image captions to be more romantic or humorous. Averaged across these three datasets, our simple baseline generated grammatical sentences with appropriate content and attribute 23% of the time, according to human raters; in contrast, the best adversarial method achieved only 12%. Our best neural system in turn outperformed our baseline, achieving an average success rate of 34%. Our code and data, including newly collected human reference outputs for the Yelp and Amazon domains, can be found at `https://github.com/lijuncen/Sentiment-and-Style-Transfer`.

2 Problem Statement

We assume access to a corpus of labeled sentences $\mathcal{D} = \{(x_1, v_1), \ldots, (x_m, v_m)\}$, where x_i is a sentence and $v_i \in \mathcal{V}$, the set of possible attributes (e.g., for sentiment, $\mathcal{V} = \{\text{"positive"}, \text{"negative"}\}$). We define $\mathcal{D}_v = \{x : (x, v) \in \mathcal{D}\}$, the set of sentences in the corpus with attribute v. Crucially, we do not assume access to a parallel corpus that pairs sentences with different attributes and the same content.

Our goal is to learn a model that takes as input (x, v^{tgt}) where x is a sentence exhibiting source (original) attribute v^{src}, and v^{tgt} is the target attribute, and outputs a sentence y that retains the content of x while exhibiting v^{tgt}.

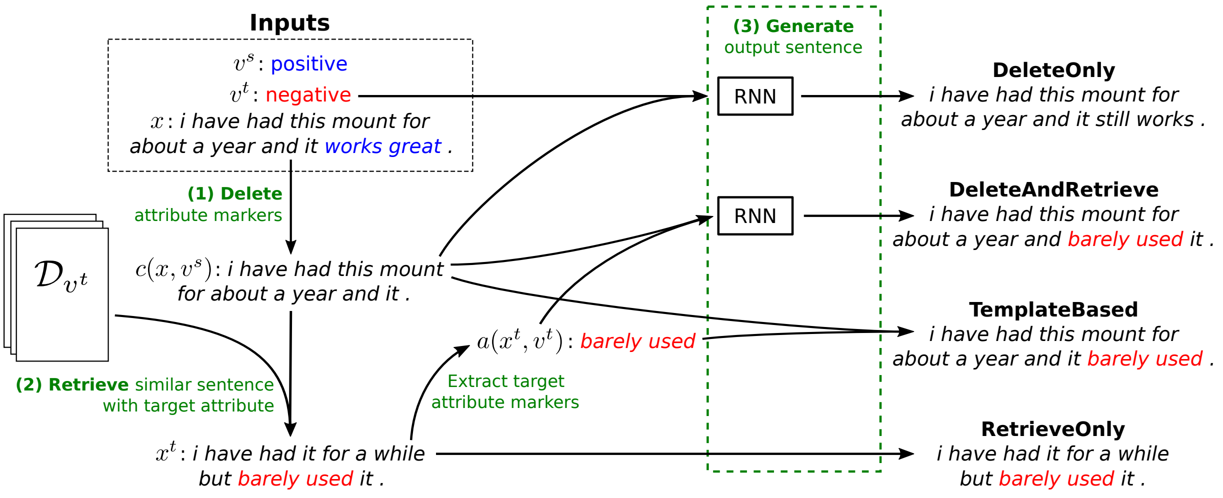
3 Approach

As a motivating example, suppose we wanted to change the sentiment of *"The chicken was delicious."* from positive to negative. Here the word *"delicious"* is the only sentiment-bearing word, so we just need to replace it with an appropriate negative sentiment word. More generally, we find that the attribute is often *localized* to a small fraction of the words, an inductive bias not captured by previous work.

How do we know which negative sentiment word to insert? The key observation is that the remaining content words provide strong cues: given *"The chicken was ... "*, one can infer that a taste-related word like *"bland"* fits, but a word like *"rude"* does not, even though both have negative sentiment. In other words, while the deleted sentiment words do contain non-sentiment information too, this information can often be recovered using the other content words.

In the rest of this section, we describe our four systems: two baselines (RetrieveOnly and TemplateBased) and two neural models (DeleteOnly and DeleteAndRetrieve). An overview of all four systems is shown in Figure 2. Formally, the main components of these systems are as follows:

1. **Delete**: All 4 systems use the same procedure to separate the words in x into a set of



Inputs
v^s: positive
v^t: negative
x: i have had this mount for about a year and it works great .
(1) Delete attribute markers
$\mathcal{D}_{v^t}$
$c(x, v^s)$: i have had this mount for about a year and it .
(2) Retrieve similar sentence with target attribute
x^t: i have had it for a while but barely used it .
$a(x^t, v^t)$: barely used
Extract target attribute markers
(3) Generate output sentence
RNN
RNN
DeleteOnly
i have had this mount for about a year and it still works .
DeleteAndRetrieve
i have had this mount for about a year and barely used it .
TemplateBased
i have had this mount for about a year and it barely used .
RetrieveOnly
i have had it for a while but barely used it .

Figure 2: Our four proposed methods on the same sentence, taken from the AMAZON dataset. Every method uses the same procedure (1) to separate attribute and content by deleting attribute markers; they differ in the construction of the target sentence. RETRIEVEONLY directly returns the sentence retrieved in (2). TEMPLATEBASED combines the content with the target attribute markers in the retrieved sentence by slot filling. DELETEANDRETRIEVE generates the output from the content and the retrieved target attribute markers with an RNN. DELETEONLY generates the output from the content and the target attribute with an RNN.

attribute markers $a(x, v^{\text{src}})$ and a sequence of content words $c(x, v^{\text{src}})$.

2. **Retrieve**: 3 of the 4 systems look through the corpus and retrieve a sentence x^{tgt} that has the target attribute v^{tgt} and whose content is similar to that of x.

3. **Generate**: Given the content $c(x, v^{\text{src}})$, target attribute v^{tgt}, and (optionally) the retrieved sentence x^{tgt}, each system generates y, either in a rule-based fashion or with a neural sequence-to-sequence model.

We describe each component in detail below.

3.1 Delete

We propose a simple method to delete attribute markers (n-grams) that have the most discriminative power. Formally, for any $v \in \mathcal{V}$, we define the *salience* of an n-gram u with respect to v by its (smoothed) relative frequency in $\mathcal{D}_v$:

$$s(u, v) = \frac{\text{count}(u, \mathcal{D}_v) + \lambda}{\left(\sum_{v' \in \mathcal{V}, v' \neq v} \text{count}(u, \mathcal{D}_{v'})\right) + \lambda}, \tag{1}$$

where $\text{count}(u, \mathcal{D}_v)$ denotes the number of times an n-gram u appears in $\mathcal{D}_v$, and λ is the smoothing parameter. We declare u to be an attribute marker for v if $s(u, v)$ is larger than a specified threshold γ. The attributed markers can be viewed as discriminative features for a Naive Bayes classifier.

We define $a(x, v^{\text{src}})$ to be the set of all source attribute markers in x, and define $c(x, v^{\text{src}})$ as the sequence of words after deleting all markers in $a(x, v^{\text{src}})$ from x. For example, for *"The chicken was delicious,"* we would delete *"delicious"* and consider *"The chicken was. . . "* to be the content (Figure 2, Step 1).

3.2 Retrieve

To decide what words to insert into $c(x, v^{\text{src}})$, one useful strategy is to look at similar sentences with the target attribute. For example, negative sentences that use phrases similar to *"The chicken was. . . "* are more likely to contain *"bland"* than *"rude."* Therefore, we retrieve sentences of similar content and use target attribute markers in them for insertion.

Formally, we retrieve x^{tgt} according to:

$$x^{\text{tgt}} = \underset{x' \in \mathcal{D}_{v^{\text{tgt}}}}{\text{argmin}}\, d(c(x, v^{\text{src}}), c(x', v^{\text{tgt}})), \tag{2}$$

where d may be any distance metric comparing two sequences of words. We experiment with two options: (i) TF-IDF weighted word overlap and (ii) Euclidean distance using the content embeddings in Section 3.3 (Figure 2, Step 2).

3.3 Generate

Finally, we describe how each system generates y (Figure 2, Step 3).

RETRIEVEONLY returns the retrieved sentence x^{tgt} verbatim. This is guaranteed to produce a grammatical sentence with the target attribute, but its content might not be similar to x.

TEMPLATEBASED replaces the attribute markers deleted from the source sentence $a(x, v^{\text{src}})$ with those of the target sentence $a(x^{\text{tgt}}, v^{\text{tgt}})$.[2] This strategy relies on the assumption that if two attribute markers appear in similar contexts , they are roughly syntactically exchangeable. For example, *"love"* and *"don't like"* appear in similar contexts (e.g., *"i **love** this place."* and *"i **don't like** this place."*), and exchanging them is syntactically valid. However, this naive swapping of attribute markers can result in ungrammatical outputs.

DELETEONLY first embeds the content $c(x, v^{\text{src}})$ into a vector using an RNN. It then concatenates the final hidden state with a learned embedding for v^{tgt}, and feeds this into an RNN decoder to generate y. The decoder attempts to produce words indicative of the source content and target attribute, while remaining fluent.

DELETEANDRETRIEVE is similar to DELETEONLY, but uses the attribute markers of the retrieved sentence x^{tgt} rather than the target attribute v^{tgt}. Like DELETEONLY, it encodes $c(x, v^{\text{src}})$ with an RNN. It then encodes the sequence of attribute markers $a(x^{\text{tgt}}, v^{\text{tgt}})$ with another RNN. The RNN decoder uses the concatenation of this vector and the content embedding to generate y.

DELETEANDRETRIEVE combines the advantages of TEMPLATEBASED and DELETEONLY. Unlike TEMPLATEBASED, DELETEANDRETRIEVE can pick a better place to insert the given attribute markers, and can add or remove function words to ensure grammaticality. Compared to DELETEONLY, DELETEANDRETRIEVE has a stronger inductive bias towards using target attribute markers that are likely to fit in the current context. Guu et al. (2018) showed that retrieval strategies like ours can help neural generative models. Finally, DELETEANDRETRIEVE gives us finer control over the output; for example, we can control the degree of sentiment by deciding whether to add *"good"* or *"fantastic"* based on the retrieved sentence x^{tgt}.

[2] Markers are replaced from left to right, in order. If there are not enough markers in x^{tgt}, we use an empty string.

3.4 Training

We now describe how to train DELETEANDRETRIEVE and DELETEONLY. Recall that at training time, we do not have access to ground truth outputs that express the target attribute. Instead, we train DELETEONLY to reconstruct the sentences in the training corpus given their content and *original* attribute value by maximizing:

$$L(\theta) = \sum_{(x, v^{\text{src}}) \in \mathcal{D}} \log p(x \mid c(x, v^{\text{src}}), v^{\text{src}}); \theta). \tag{3}$$

For DELETEANDRETRIEVE, we could similarly learn an auto-encoder that reconstructs x from $c(x, v^{\text{src}})$ and $a(x, v^{\text{src}})$. However, this results in a trivial solution: because $a(x, v^{\text{src}})$ and $c(x, v^{\text{src}})$ were known to come from the same sentence, the model merely learns to stitch the two sequences together without any smoothing. Such a model would fare poorly at test time, when we may need to alter some words to fluently combine $a(x^{\text{tgt}}, v^{\text{tgt}})$ with $c(x, v^{\text{src}})$. To address this train/test mismatch, we adopt a denoising method similar to the denoising auto-encoder (Vincent et al., 2008). During training, we apply some noise to $a(x, v^{\text{src}})$ by randomly altering each attribute marker in it independently with probability 0.1. Specifically, we replace an attribute marker with another randomly selected attribute marker of the same attribute and word-level edit distance 1 if such a noising marker exists, e.g., *"was very rude"* to *"very rude"*, which produces $a'(x, v^{\text{src}})$.

Therefore, the training objective for DELETEANDRETRIEVE is to maximize:

$$L(\theta) = \sum_{(x, v^{\text{src}}) \in \mathcal{D}} \log p(x \mid c(x, v^{\text{src}}), a'(x, v^{\text{src}}); \theta). \tag{4}$$

4 Experiments

We evaluated our approach on three domains: flipping sentiment of Yelp reviews (YELP) and Amazon reviews (AMAZON), and changing image captions to be romantic or humorous (CAPTIONS). We compared our four systems to human references and three previously published adversarial approaches. As judged by human raters, both of our two baselines outperform all three adversarial methods. Moreover, DELETEANDRETRIEVE outperforms all other automatic approaches.

Dataset	Attributes	Train	Dev	Test
YELP	Positive	270K	2000	500
	Negative	180K	2000	500
CAPTIONS	Romantic	6000	300	0
	Humorous	6000	300	0
	Factual	0	0	300
AMAZON	Positive	277K	985	500
	Negative	278K	1015	500

Table 1: Dataset statistics.

4.1 Datasets

First, we describe the three datasets we use, which are commonly used in prior works too. All datasets are randomly split into train, development, and test sets (Table 1).

YELP Each example is a sentence from a business review on Yelp, and is labeled as having either positive or negative sentiment.

AMAZON Similar to YELP, each example is a sentence from a product review on Amazon, and is labeled as having either positive or negative sentiment (He and McAuley, 2016).

CAPTIONS In the CAPTIONS dataset (Gan et al., 2017), each example is a sentence that describes an image, and is labeled as either factual, romantic, or humorous. We focus on the task of converting factual sentences into romantic and humorous ones. Unlike YELP and AMAZON, CAPTIONS is actually an aligned corpus—it contains captions for the same image in different styles. Our systems do not use these alignments, but we use them as gold references for evaluation.

CAPTIONS is also unique in that we reconstruct romantic and humorous sentences during training, whereas at test time we are given factual captions. We assume these factual captions carry only content, and therefore do not look for and delete factual attribute markers; The model essentially only inserts romantic or humorous attribute markers as appropriate.

4.2 Human References

To supply human reference outputs to which we could compare the system outputs for YELP and AMAZON, we hired crowdworkers on Amazon Mechanical Turk to write gold outputs for all test sentences. Workers were instructed to edit a sentence to flip its sentiment while preserving its content.

Our delete-retrieve-generate approach relies on the prior knowledge that to accomplish attribute transfer, a small number of attribute markers should be changed, and most other words should be kept the same. We analyzed our human reference data to understand the extent to which humans follow this pattern. We measured whether humans preserved words our system marks as content, and changed words our system marks as attribute-related (Section 3.1). We define the *content word preservation rate* S_c as the average fraction of words our system marks as content that were preserved by humans, and the *attribute-related word change rate* S_a as the average fraction of words our system marks as attribute-related that were changed by humans:

$$S_c = \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{(x, v^{\text{src}}, y^*) \in \mathcal{D}_{\text{test}}} \frac{|c(x, v^{\text{src}}) \cap y^*|}{|c(x, v^{\text{src}})|}$$
$$S_a = 1 - \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{(x, v^{\text{src}}, y^*) \in \mathcal{D}_{\text{test}}} \frac{|a(x, v^{\text{src}}) \cap y^*|}{|a(x, v^{\text{src}})|}, \quad (5)$$

where $\mathcal{D}_{\text{test}}$ is the test set, y^* is the human reference sentence, and $| \cdot |$ denotes the number of non-stopwords. Higher values of S_c and S_a indicate that humans preserve content words and change attribute-related words, in line with the inductive bias of our model. S_c is 0.61, 0.71, and 0.50 on YELP, AMAZON, and CAPTIONS, respectively; S_a is 0.72 on YELP and 0.54 on AMAZON (not applicable on CAPTIONS).

To understand why humans sometimes deviated from the inductive bias of our model, we randomly sampled 50 cases from YELP where humans changed a content word or preserved an attribute-related word. 70% of changed content words were unimportant words (e.g., *"whole"* was deleted from *"whole experience"*), and another 18% were paraphrases (e.g., *"charge"* became *"price"*); the remaining 12% were errors where the system mislabeled an attribute-related word as a content word (e.g., *"old"* became *"new"*). 84% of preserved attribute-related words did pertain to sentiment but remained fixed due to changes in the surrounding context (e.g., *"don't **like**"* became *"**like**"*, and *"below **average**"* became *"above **average**"*); the remaining 16% were mistagged by our system as being attribute-related (e.g., *"walked out"*).

4.3 Previous Methods

We compare with three previous models, all of which use adversarial training. **STYLEEMBED-**

	YELP				AMAZON				CAPTIONS			
	Gra	Con	Att	Suc	Gra	Con	Att	Suc	Gra	Con	Att	Suc
CROSSALIGNED	2.8	2.9	3.5	14%	3.2	2.5	2.9	7%	3.9	2.0	3.2	16%
STYLEEMBEDDING	3.5	3.7	2.1	9%	3.2	2.9	2.8	11%	3.3	2.9	3.0	17%
MULTIDECODER	2.8	3.1	3.0	8%	3.0	2.6	2.8	7%	3.4	2.8	3.2	18%
RETRIEVEONLY	**4.2**	2.7	**4.2**	25%	3.8	2.8	3.1	17%	**4.2**	2.6	3.8	27%
TEMPLATEBASED	3.0	**3.9**	3.9	21%	3.4	3.6	3.1	19%	3.3	**4.1**	3.5	33%
DELETEONLY	3.0	3.7	3.9	24%	3.7	**3.8**	3.2	24%	3.6	3.5	3.5	32%
DELETEANDRETRIEVE	3.3	3.7	4.0	**29%**	**3.9**	3.7	**3.4**	**29%**	3.8	3.5	**3.9**	**43%**
Human	4.6	4.5	4.5	75%	4.2	4.0	3.7	44%	4.3	3.9	4.0	56%

Table 2: Human evaluation results on all three datasets. We show average human ratings for grammaticality (Gra), content preservation (Con), and target attribute match (Att) on a 1 to 5 Likert scale, as well as overall success rate (Suc). On all three datasets, DELETEANDRETRIEVE is the best overall system, and all four of our methods outperform previous work.

DING (Fu et al., 2018) learns an vector encoding of the source sentence such that a decoder can use it to reconstruct the sentence, but a discriminator, which tries to identify the source attribute using this encoding, fails. They use a basic MLP discriminator and an LSTM decoder. **MULTIDECODER** (Fu et al., 2018) is similar to STYLEEMBEDDING, except that it uses a different decoder for each attribute value. **CROSSALIGNED** (Shen et al., 2017) also encodes the source sentence into a vector, but the discriminator looks at the hidden states of the RNN decoder instead. The system is trained so that the discriminator cannot distinguish these hidden states from those obtained by forcing the decoder to output real sentences from the target domain; this objective encourages the real and generated target sentences to look similar at a population level.

4.4 Experimental Details

For our methods, we use 128-dimensional word vectors and a single-layer GRU with 512 hidden units for both encoders and the decoder. We use the maxout activation function (Goodfellow et al., 2013). All parameters are initialized by sampling from a uniform distribution between -0.1 and 0.1. For optimization, we use Adadelta (Zeiler, 2012) with a minibatch size of 256.

For attribute marker extraction, we consider spans up to 4 words, and the smoothing parameter λ is set to 1. We set the attribute marker threshold γ, which controls the precision and recall of our attribute markers, to 15, 5.5 and 5 for YELP, AMAZON, and CAPTIONS. These values were set by manual inspection of the resulting markers and tuning slightly on the dev set. For retrieval, we used the TF-IDF weighted word overlap score for DELETEANDRETRIEVE and TEMPLATEBASED, and the Euclidean distance of content embeddings for RETRIEVEONLY. We find the two scoring functions give similar results.

For all neural models, we do beam search with a beam size of 10. For DELETEANDRETRIEVE, similar to Guu et al. (2018), we retrieve the top-10 sentences and generate results using markers from each sentence. We then select the output with the lowest perplexity given by a separately-trained neural language model on the target-domain training data.

4.5 Human Evaluation

We hired workers on Amazon Mechanical Turk to rate the outputs of all systems. For each source sentence and target attribute, the same worker was shown the output of each tested system. Workers were asked to rate each output on three criteria on a Likert scale from 1 to 5: grammaticality, similarity to the target attribute, and preservation of the source content. Finally, we consider a generated output "successful" if it is rated 4 or 5 on all three criteria. For each dataset, we evaluated 400 randomly sampled examples (200 for each target attribute).

Table 2 shows the human evaluation results. On all three datasets, both of our baselines have a higher success rate than the previously published models, and DELETEANDRETRIEVE achieves the best performance among all systems. Additionally, we see that human raters strongly preferred the human references to all systems, suggesting there is still significant room for improvement on this task.

We find that a human evaluator's judgment of a sentence is largely relative to other sentences being evaluated together and examples given in the instruction (different for each dataset/task). Therefore, evaluating all system outputs in one batch is important and results on different datasets are not directly comparable.

4.6 Analysis

We analyze the strengths and weaknesses of the different systems. Table 3 show typical outputs of each system on the YELP and CAPTIONS dataset.

We first analyze the adversarial methods. CROSSALIGNED and MULTIDECODER tend to lose the content of the source sentence, as seen in both the example outputs and the overall human ratings. The decoder tends to generate a frequent but only weakly related sentence with the target attribute. On the other hand, STYLEEMBEDDING almost always generates a paraphrase of the input sentence, implying that the encoder preserves some attribute information. We conclude that there is a delicate balance between preserving the original content and dropping the original attribute, and existing adversarial models tend to sacrifice one or the other.

Next, we analyze our baselines. RETRIEVEONLY scores well on grammaticality and having the target attribute, since it retrieves sentences with the desired attribute directly from the corpus. However, it is likely to change the content when there is no perfectly aligned sentence in the target domain. In contrast, TEMPLATEBASED is good at preserving the content because the content words are guaranteed to be kept. However, it makes grammatical mistakes due to the unsmoothed combination of content and attribute words.

DELETEANDRETRIEVE and DELETEONLY achieve a good balance among grammaticality, preserving content, and changing the attribute. Both have strong inductive bias on what words should be changed, but still have the flexibility to smooth out the sentence. The main difference is that DELETEONLY fills in attribute words based on only the target attribute, whereas DELETEANDRETRIEVE conditions on retrieved attribute words. When there is a diverse set of phrases to fill in—for example in CAPTIONS— conditioning on retrieved attribute words helps generate longer sentences with more specific attribute descriptions.

4.7 Automatic Evaluation

Following previous work (Hu et al., 2017; Shen et al., 2017), we also compute automatic evaluation metrics, and compare these numbers to our human evaluation results.

We use an attribute classifier to assess whether outputs have the desired attribute (Hu et al., 2017; Shen et al., 2017). We define the *classifier score* as the fraction of outputs classified as having the target attribute. For each dataset, we train an attribute classifier on the same training data. Specifically, we encode the sentence into a vector by a bidirectional LSTM with an average pooling layer over the outputs, and train the classifier by minimizing the logistic loss.

We also compute BLEU between the output and the human references, similar to Gan et al. (2017). A high BLEU score primarily indicates that the system can correctly preserve content by retaining the same words from the source sentence as the reference. One might also hope that it has some correlation with fluency, though we expect this correlation to be much weaker.

Table 4 shows the classifier and BLEU scores. In Table 5, we compute the system-level correlation between classifier score and human judgments of attribute transfer, and between BLEU and human judgments of content preservation and grammaticality. We also plot scores given by the automatic metrics and humans in Figure 4. While the scores are sometimes well-correlated, the results vary significantly between datasets; on AMAZON, there is no correlation between the classifier score and the human evaluation. Manual inspection shows that on AMAZON, some product genres are associated with either mostly positive or mostly negative reviews. However, our systems produce, for example, negative reviews about products that are mostly discussed positively in the training set. Therefore, the classifier often gives unreliable predictions on system outputs. As expected, BLEU does not correlate well with human grammaticality ratings. The lack of automatic fluency evaluation artificially favors systems like TEMPLATEBASED, which make more grammatical mistakes. We conclude that while these automatic evaluation methods are useful for model development, they cannot replace human evaluation.

4.8 Trading off Content versus Attribute

One advantage of our methods is that we can control the trade-off between matching the target attribute and preserving the source content. To achieve different points along this trade-off curve,

	From negative to positive (YELP)
SOURCE	we sit down and we got some really slow and lazy service .
CROSSALIGNED	we *went* down and we *were a good , friendly food* .
STYLEEMBEDDING	we sit down and we got some really *slow* and *prices suck* .
MULTIDECODER	we sit down and we got some really and *fast food* .
TEMPLATEBASED	we sit down and we got some *the service is always great* and *even better* service .
RETRIEVEONLY	*i* got *a veggie hoagie that was massive* and some *grade a customer* service .
DELETEONLY	we sit down and we got some *great* and *quick* service .
DELETEANDRETRIEVE	we got *very nice place to* sit down and we got some service .
	From factual to romantic (CAPTIONS)
SOURCE	two dogs play by a tree .
CROSSALIGNED	*a dog is running through the grass* .
STYLEEMBEDDING	two dogs play *against* a tree .
MULTIDECODER	two dogs play by a tree .
TEMPLATEBASED	two dogs play by a tree *loving* .
RETRIEVEONLY	two dogs *are playing in a pool as best friends* .
DELETEANDRETRIEVE	two dogs play by a tree , *enjoying the happiness of childhood* .
DELETEONLY	two dogs *in love* play *happily* by a tree .
	From negative to positive (AMAZON)
SOURCE	this is the worst game i have come across in a long time .
CROSSALIGNED	this is the *best thing* i *ve had for a few years* .
STYLEEMBEDDING	this is the worst game i have come across in a long time .
MULTIDECODER	this is the *best knife* i have *no room with* a long time .
TEMPLATEBASED	this is the *best* come across in a long time .
RETRIEVEONLY	*the customer support is some of the best* i have come across in a long time .
DELETEONLY	this is the *best* game i have come across in a long time .
DELETEANDRETRIEVE	this is the *best* game i have come across in a long time .

Table 3: Example outputs on YELP, CAPTIONS, and AMAZON. Additional examples for transfer from opposite directions are given in Table 6. Added or changed words are in *italic*. Attribute markers are colored.

	YELP		CAPTIONS		AMAZON	
	Classifier	BLEU	Classifier	BLEU	Classifier	BLEU
CROSSALIGNED	73.7%	3.1	74.3%	0.1	**74.1%**	0.4
STYLEEMBEDDING	8.7%	**11.8**	54.7%	6.7	43.3%	10.0
MULTIDECODER	47.6%	7.1	68.5%	4.6	68.3%	5.0
TEMPLATEBASED	81.7%	**11.8**	92.5%	**17.1**	68.7%	**27.1**
RETRIEVEONLY	**95.4%**	0.4	**95.5%**	0.7	**70.3%**	0.9
DELETEONLY	85.7%	7.5	83.0%	9.0	45.6%	24.6
DELETEANDRETRIEVE	88.7%	8.4	**96.8%**	7.3	48.0%	22.8

Table 4: Automatic evaluation results. "Classifier" shows the percentage of sentences labeled as the target attribute by the classifier. BLEU measures content similarity between the output and the human reference.

we simply vary the threshold γ (Section 3.1) *at test time* to control how many attribute markers we delete from the source sentence. In contrast, other methods (Shen et al., 2017; Fu et al., 2018) would require retraining the model with different hyperparameters to achieve this effect.

Figure 3 shows this trade-off curve for DELETEANDRETRIEVE, DELETEONLY, and TEMPLATEBASED on YELP, where target attribute match is measured by the classifier score and content preservation is measured by BLEU.[3] We see a clear trade-off between changing the attribute and retaining the content.

[3] RETRIEVEONLY is less affected by what content words are preserved, especially when no good output sentence exists in the target corpus. Therefore, we found that it did not exhibit a clear content-attribute trade-off.

5 Related Work and Discussion

Our work is closely related to the recent body of work on text attribute transfer with *unaligned* data, where the key challenge to disentangle attribute and content in an unsupervised way. Most existing work (Shen et al., 2017; Zhao et al., 2018; Fu et al., 2018; Melnyk et al., 2017) uses adversarial training to separate attribute and content: the content encoder aims to fool the attribute discriminator by removing attribute information from the content embedding. However, we find that empirically it is often easy to fool the discriminator without actually removing the attribute information. Therefore, we explicitly separate attribute and content by taking advantage of the prior knowledge that the attribute is localized to parts of the sentence.

To address the problem of unaligned data, Hu

	Classifier	BLEU	
	Attribute	Content	Grammaticality
All data	0.810 ($p < 0.01$)	0.876 ($p < 0.01$)	-0.127 ($p = 0.58$)
YELP	0.991 ($p < 0.01$)	0.935 ($p < 0.01$)	0.119 ($p = 0.80$)
CAPTIONS	0.982 ($p < 0.01$)	0.991 ($p < 0.01$)	-0.631 ($p = 0.13$)
AMAZON	-0.036 ($p = 0.94$)	0.857 ($p < 0.01$)	0.306 ($p = 0.50$)

Table 5: Spearman correlation between two automatic evaluation metrics and related human evaluation scores. While some correlations are strong, the classifier exhibits poor correlation on AMAZON, and BLEU only measures content, not grammaticality.

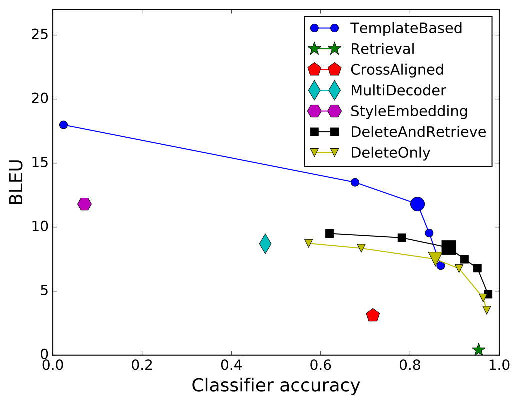


Figure 3: Trade-off curves between matching the target attribute (measured by classifier scores) and preserving the content (measured by BLEU). Bigger points on the curve correspond to settings used for both training and our official evaluation.

et al. (2017) relies on an attribute classifier to guide the generator to produce sentences with a desired attribute (e.g. sentiment, tense) in the Variational Autoencoder (VAE) framework. Similarly, Zhao et al. (2018) used a regularized autoencoder in the adversarial training framework; however, they also find that these models require extensive hyperparameter tuning and the content tends to be changed during the transfer. Shen et al. (2017) used a discriminator to align target sentences and sentences transfered to the target domain from the source domain. More recently, unsupervised machine translation models (Artetxe et al., 2017; Lample et al., 2017) used a cycle loss similar to Jun-Yan et al. (2017) to ensure that the content is preserved during the transformation. These methods often rely on bilingual word vectors to provide word-for-word translations, which are then finetune by back-translation. Thus they can be used to further improve our results.

Our method of detecting attribute markers is reminiscent of Naive Bayes, which is a strong baseline for tasks like sentiment classification (Wang and Manning, 2012). Deleting these attribute markers can be viewed as attacking a Naive Bayes classifier by deleting the most informative features (Globerson and Roweis, 2006), similarly to how adversarial methods are trained to fool an attribute classifier. One difference is that our classifier is fixed, not jointly trained with the model.

To conclude, we have described a simple method for text attribute transfer that outperforms previous models based on adversarial training. The main leverage comes from the inductive bias that attributes are usually manifested in localized discriminative phrases. While many prior works on linguistic style analysis confirm our observation that attributes often manifest in idiosyncratic phrases (Recasens et al., 2013; Schwartz et al., 2017; Newman et al., 2003), we recognize the fact that in some problems (e.g., Pavlick and Tetreault (2017)), content and attribute cannot be so cleanly separated along phrase boundaries. Looking forward, a fruitful direction is to develop a notion of attributes more general than n-grams, but with more inductive bias than arbitrary latent vectors.

Reproducibility. All code, data, and experiments for this paper are available on the CodaLab platform at `https://worksheets.codalab.org/worksheets/0xe3eb416773ed4883bb737662b31b4948/`.

Acknowledgements. This work is supported by the DARPA Communicating with Computers (CwC) program under ARO prime contract no. W911NF- 15-1-0462. J.L. is supported by Tencent. R.J. is supported by an NSF Graduate Research Fellowship under Grant No. DGE-114747.

References

M. Arjovsky and L. Bottou. 2017. Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representations (ICLR)*.

M. Artetxe, G. Labaka, E. Agirre, and K. Cho. 2017.

Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041* .

K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. 2017. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Computer Vision and Pattern Recognition (CVPR)*.

Z. Fu, X. Tan, N. Peng, D. Zhao, and R. Yan. 2018. Style transfer in text: Exploration and evaluation. In *Association for the Advancement of Artificial Intelligence (AAAI)*.

C. Gan, Z. Gan, X. He, J. Gao, and L. Deng. 2017. Stylenet: Generating attractive visual captions with styles. In *Computer Vision and Pattern Recognition (CVPR)*.

A. Globerson and S. Roweis. 2006. Nightmare at test time: robust learning by feature deletion. In *International Conference on Machine Learning (ICML)*. pages 353–360.

I. Goodfellow, D. Warde-farley, M. Mirza, A. Courville, and Y. Bengio. 2013. Maxout networks. In *International Conference on Machine Learning (ICML)*. pages 1319–1327.

K. Guu, T. B. Hashimoto, Y. Oren, and P. Liang. 2018. Generating sentences by editing prototypes. *Transactions of the Association for Computational Linguistics (TACL)* 0.

R. He and J. McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *World Wide Web (WWW)*.

Z. Hu, Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing. 2017. Toward controlled generation of text. In *International Conference on Machine Learning (ICML)*.

Z. Jun-Yan, P. Taesung, I. Phillip, and E. A. A. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *International Conference on Computer Vision (ICCV)*.

G. Lample, L. Denoyer, and M. Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043* .

I. Melnyk, C. N. dos Santos, K. Wadhawan, I. Padhi, and A. Kumar. 2017. Improved neural text attribute transfer with non-parallel data. *arXiv preprint arXiv:1711.09395* .

M. L. Newman, J. W. Pennebaker, D. S. Berry, and J. M. Richards. 2003. Lying words: Predicting deception from linguistic styles. *Personality and Social Psychology Bulletin* 29.

E. Pavlick and J. Tetreault. 2017. An empirical analysis of formality in online communication. *Transactions of the Association for Computational Linguistics (TACL)* 4.

M. Recasens, C. Danescu-Niculescu-Mizil, and D. Jurafsky. 2013. Linguistic models for analyzing and detecting biased language. In *Association for Computational Linguistics (ACL)*.

T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. 2016. Improved techniques for training gans. In *Advances in Neural Information Processing Systems (NIPS)*.

R. Schwartz, M. Sap, Y. Konstas, L. Zilles, Y. Choi, and N. A. Smith. 2017. The effect of different writing tasks on linguistic style: A case study of the ROC story cloze task. In *Computational Natural Language Learning (CoNLL)*.

T. Shen, T. Lei, R. Barzilay, and T. Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems (NIPS)*.

P. Vincent, H. Larochelle, Y. Bengio, , and P. Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *International Conference on Machine Learning (ICML)*.

S. Wang and C. D. Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Association for Computational Linguistics (ACL)*.

M. D. Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701* .

J. Zhao, Y. Kim, K. Zhang, A. M. Rush, and Y. LeCun. 2018. Adversarially regularized autoencoders. In *International Conference on Learning Representations (ICLR)*.