# Transforming Chatbot Responses to Mimic Domain-specific Linguistic Styles

Siddhartha Banerjee[1]**, Prakhar Biyani[2], and Kostas Tsioutsiouliklis[2]

[1] The Pennsylvania State University, University Park, PA, USA
sbanerjee@ist.psu.edu,
[2] Yahoo!, Sunnyvale, California, USA,
{pxb5080, kostas}@yahoo-inc.com

**Abstract.** Chatbots and conversational agents have become very popular in recent years and there is a huge research effort to automate conversations in several applications. Even if a bot provides accurate answers, users generally have a better experience if the chatbots can mimic certain personalities that users are acquainted with. In this work, we make an attempt to transform regular chatbot responses to produce domain-specific responses that can mimic speaking styles uniquely associated with a particular domain (community of similar personalities such as *politicians*, *singers*, etc.). We construct domain-specific word-graphs using tweets posted from Twitter accounts that belong to users from specific domains and use the graph to generate word-patterns. New words (obtained from the patterns in the graph) are introduced to transform the regular responses. We prune the graph using data-driven thresholds such as co-occurrence metrics to avoid spurious transformations. Furthermore, we use paragraph vectors to re-rank generated patterns and use only the patterns that are contextually similar to the original response. Our initial analysis shows that generated patterns from different domains show marked differences in style.

**Keywords:** chatbots, personality, community, speaking style, language generation

## 1 Introduction

With the tremendous growth in the field of Artificial Intelligence (AI), chatbots[3] have become very popular and there is a growing interest in building end-to-end conversational systems. However, there has not been much work on generating responses to mimic specific speaking styles of personalities. Li et. al, [9] describe a persona-based neural conversation model where, the *persona* is restricted to general human-like behavior and not specific persona styles. In this work, our focus is to provide an accurate answer to questions asked to the conversational agent. Simultaneously, our goal is to develop a system that can transform regular chatbot responses to mimic styles of specific domains with which users can relate and are acquainted. For example, a user interested in *fashion* or *entertainment* would enjoy getting bot responses resembling the speaking styles of fashionistas or entertainers, respectively.

** This work was done during an internship at Yahoo!

[3] https://en.wikipedia.org/wiki/Chatterbot

We assume that a regular chatbot response (without any stylistic elements) is provided. We restrict our work to two domains – *politics* and *entertainment* in this paper. The transformation should retain the factual content of the response but add a distinctive style such that one can easily identify and attribute the response to a specific domain. Consider the following example where a chatbot is requested information on today's weather. The responses as expected from the two domains (politics and entertainment) are shown in the output. Note that the following responses are not machine-generated.

> **Normal output:** It is very hot today.
> **Output (politics):** Ladies and gentlemen, it appears to be very hot today.
> **Output (entertainment):** Brace yourselves, it's kinda hot today!

Twitter[4] users constitute different types of personas such as politicians, singers, actors, sports persons, etc.. Therefore, we use tweets as our data source to model domain-specific styles. Identifying differences in vocabularies and word-usage patterns across domains is critical in modeling domain peculiarities and hence, differentiating between domains. For examples, tweets from fashionistas contain informal language (*xoxo*, *ahhhhh*) and heavy usage of emoticons. In contrast, tweets from politicians are more formal. If the peculiarities in the domain-specific style can be introduced in the response, keeping its existing factual content intact, we can possibly mimic the style of a specific domain. This intuition forms the core of our methodology. We construct two separate word-graphs using tweets from Twitter handles (accounts) belonging to the two domains. We adapt the word-graph construction from Filippova's multi-sentence compression approach [6], where the nodes represent words (along with part-of-speech (POS) tags) and the edges connect two adjacent words. However, we extend the approach to construct a reliable graph by ignoring edges and nodes which do not meet specific constraints. In other words, the infrequent edges in the set of tweets are removed. Traversing the word-graph from one node to another results in several paths which form certain word-patterns. We filter out patterns containing nouns to prevent deviation from the actual information. Furthermore, we also restrict paths between pairs where the second word is an auxiliary verb to avoid introducing irrelevant patterns. We rank all the generated patterns with respect to the context of the initial regular response to avoid vague and arbitrary responses. Specifically, we use semantic similarity between the original response and generated paths using distributed representations [8] to rank the paths and retain only top few paths.

We conduct several experiments to evaluate the effectiveness of our approach. To start with, we manually frame some factual sentences which do not contain any stylistic elements and then generate transformed responses for those sentences using our approach. We train graphs using tweets from two domains – *politics* and *entertainment*. We use thresholds for pruning the word-graph and selecting contextual responses from a ranked list of generated responses. Our initial analysis of the generated responses shows that our technique can transform a basic response to a domain-specific stylistic response and, hence has a strong potential to mimic domain-specific styles.

[4] `https://twitter.com/`

## 2 Related Work

Conversational agents have received major attention from researchers, especially from the perspective of Natural Language Generation (NLG). Receiving accurate responses from a chatbot is essential; however, bots that can mimic personas or specific speaking styles can help in user retention. Li et. al, [9] attempted to mimic human-like conversations using a persona-model using a deep neural-network model. However, their work does not differentiate between speakers from different domains and does not generate text conforming to specific styles as evident from the examples shown in their work. In contrast to their work, we do not aim to generate responses in conversations but modify regular responses to fit domain-specific styles. Personas of film characters have been studied recently [1] but there is no language generation involved in their work. Duplessis et. al, [4] created a chatterbot aimed at selecting the best possible response from a list of pre-populated responses. Several other virtual agents have been developed that select the best response [5, 3] from a set of responses. Kubon et. al [7] presented a chatterbot that generated responses similar to a politician using manually generated templates. Rushforth et. al, [13] proposed a model based on the five-factor personalities [10] to model characters; however, it was focused towards perceiving personality traits rather than persona-based content generation. In contrast to the above-mentioned approaches, our method is completely data-driven that transforms factual responses without relying on predefined responses or templates.

## 3 Proposed Approach

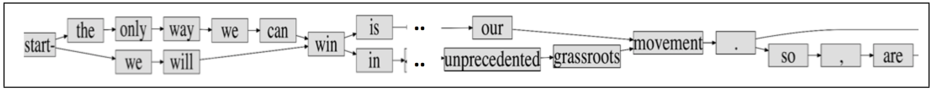
In this section, we explain the word-graph generation method, the graph pruning strategy to remove unreliable edges from the graph, and the ranking of generated patterns using semantic similarity between distributional representations of the input sentence and the generated patterns.

### 3.1 Word-graph Construction

To generate domain-specific transformations, we construct two separate word-graphs corresponding to the two domains by using tweets posted by famous personalities in those domains (See Section 4.1 for details). The word-graph is used to identify patterns of word-usages in a specific domain. We tokenize the tweets and tag the tokens with part-of-speech tags using a Twitter-specific POS tagger [11]. We do not consider retweets as they are duplicates of the original tweets. We modify all the URLs, hashtags, numbers and Twitter handles to standard tokens. For example, all URL's are changed to $< URL >$ tag. As mentioned earlier, we adapt Filippova's word-graph [6] construction technique. A node in the graph is represented by a token, which is a combination of a word and its POS tag. Tokens from a tweet are iteratively added or mapped to the graph as nodes. An edge is created between two nodes if the corresponding tokens are adjacent in the tweet. We maintain the adjacency direction between the tokens by having directed edges. The word graph is a directed acyclic graph. We describe the details later in this section.

**Mapping criteria:** We use the following set of rules to add or map a new token $t$ to the graph. If there are no nodes with the same corresponding word and POS tag as $t$, we create a new node with token $t$. If there is only one node in the graph with the same corresponding word and POS tag as $t$ then we map $t$ to

**Fig. 1.** Word-graph from two sample tweets (not all words shown to maintain clarity)



that node. In case there are multiple nodes with the same word and POS tag as $t$, we assign $t$ to the node which has the highest contextual similarity with $t$. We define contextual similarity as the number of common words within a window of one word on either side of the nodes and the current token ($t$) in the tweet. If multiple nodes have the same contextual similarity with $t$, then we assign $t$ randomly to one of those nodes. Also, if contextual similarity is zero for all the nodes, we create a new node with $t$ as token. Determining the context helps us avoid spurious mappings of words to existing nodes.

Since adjacency between two tokens across tweets can be bidirectional, we use the following strategy to maintain the acyclic nature of our graph. Lets assume we have a tweet with the following bigram $t_1\_t_2$, where $t_1$ and $t_2$ are the two tokens in the bigram. For this adjacency, we'll have a directed edge from node $n_1$ to node $n_2$ whose corresponding tokens are $t_1$ and $t_2$ respectively. If for some tweet, we get a reverse bigram, i.e, $t_2\_t_1$, then to avoid forming a cycle between nodes $n_1$ and $n_2$, we map $t_2$ to $n_2$ using the above mentioned criteria, but do not map $t_1$ to $n_1$ even if the mapping criteria are met. We either create a new node for $t_1$ or assign it to some node (other than $n_1$) depending upon whether the mapping criteria are met or not.

Fig 1 shows the word graph for the following tweets in the *politics* domain:
1. *We will win in 2016 because we are going to create an unprecedented grassroots movement.*
2. *The only way we can win is if enough people come together to join our movement. So, are you in?*

As can be seen from the figure, the tweets have common words such as *win* and *movement*. Merging the sentences along the words would result in several new possible patterns between pairs of words. For example, we can now obtain a pattern – ‘ ‘*unprecedented grassroots movement. So, are you in?*” between *unprecedented* and *in*, that did not exist in any of the two tweets but is now generated as a result of fusion between both the tweets. Two dummy nodes (*-start-* and *-end-*) are introduced to map the beginning and end of all the tweets.

**Pruning**: Constructing a word graph using adjacency relations results in a large number of edges. Not all the edges are very frequent, and might contain grammatically incorrect sequences due to the general informal style of tweets. Therefore, a significant number of such edges are irrelevant and should be removed. To favor relevant and grammatically correct word patterns, we perform *pruning* at both node and edge level. We remove the nodes from the graph that have less than 5 edges (including both outgoing and incoming edges). Also, we compute edge weights using the following equation 1 and remove edges with weights lower than the weight at $t_{perc}^{th}$ percentile value.

$$W(e_{ij}) = \frac{freq(w_i w_j)}{freq(w_i) * freq(w_j)} \tag{1}$$

In equation 1, $W(e_{ij})$ denotes the weight of edge $e_{ij}$ between nodes $i$ and $j$ with corresponding tokens $w_i$ and $w_j$ respectively, and $freq$ denotes the frequency.

Consequently, the numerator computes the frequency of co-occurrence of tokens $w_i$ and $w_j$, and the denominator computes unigram frequencies of $w_i$ and $w_j$.

### 3.2 Transformation

We transform regular responses by introducing relevant word patterns between existing words of the response without modifying its factual content (not modifying original words). The various steps involved in the transformation are explained below.

**Pattern generation**: We use the NLTK Treebank tokenizer [2] to tokenize the input response. Between each pair of words in the response, our goal is to introduce new patterns from the domain-specific word graph. We use some basic syntactic rules to improve grammatical correctness of the generated word patterns in the final output. For example, if the second word in a word pair is an auxiliary verb (such as *is*, *are*), we do not introduce any words between the pair. Without this constraint, several irrelevant words are introduced between the stopwords that result in incoherent output. We do not introduce patterns between proper nouns. Also, currently, we restrict the number of words that can be introduced between any pair of words to two. Finally, the patterns generated between each pair of words are combined to generate the pattern for the entire response. Next, we explain the pattern generation for the sentence '*'He is a loser"* using the word graph built using tweets from the entertainment domain. Following are the pairs of words between which we would introduce patterns: (i) -start- , he (ii) is, a (iii) a, loser (iv) loser, -end-

As mentioned earlier, the *-start-* and *-end-* tokens are dummy tokens used to mark the start and end of the input. As a result, patterns are also introduced before the first word and after the last word in the response. Given a particular word graph, we found several patterns between each pair of words. Some patterns with high weights are as follows: is *literally making* a, a *total* loser, loser *!!! xoxoxo* -end-. Combining all the suggested patterns, results in the following sequence: *He is literally making a total loser !!! xoxoxo.* We see that the input sentence is significantly transformed to reflect the casual writing style used in tweets from users that belong to the *entertainment* category.

**Contextually relevant pattern identification**: Generated patterns should be contextually relevant to the original input otherwise the final response may be incoherent and vague. For example, to transform *bond market*, we should include patterns that fit into the context of the *financial bond sector* and not the *bond movie.* To obtain contextually relevant patterns, we compute similarities between the original input response and the generated patterns from the word-graph. We represent both the input and the patterns as vector representations using Paragraph2Vec [8][5] and compute the cosine similarities between the input and each pattern. The patterns with higher cosine similarities are ranked higher. Responses are transformed using the top 5 patterns.

## 4 Experiments and Results

We perform qualitative evaluation of our approach by analyzing generated transformations of some regular responses. While generating relevant stylistic text is

[5] We used the implementation of paragraph vectors from the gensim [12] library.

the primary goal, the system should be able to compute the linguistic transformations fast. Therefore, we also report average run-time of transformations for the two domains. Furthermore, we also determine the thresholds of pruning the graph. In this section, first, we describe our dataset followed by the experiments on qualitative judgments of the transformations and runtime analysis for response transformations.

### 4.1 Dataset

We extracted tweets from the Twitter Firehose that were posted between November 2014 and May 2016 in the two domains – *politics* and *entertainment*. We used around $105,000$ tweets in each category to construct the word graphs. For the *politics* domain, we extracted tweets of US politicians' handles such as *realDonaldTrump, HillaryClinton, BarackObama, SenJohnMcCain, SpeakerBoehner, JoeBiden, reppaulryan,* etc. For the *entertainment* domain, we extracted tweets of celebrities involved with fashion, music etc. such as *khloekardashian, kourtneykardash, britneyspears, KylieJenner, KendallJenner.*

### 4.2 Domain-specific transformations

We generate transformations of responses using both domain-specific word-graphs (*politics* and *entertainment*). We set $t_{perc}$ to 0.5 for pruning the graphs of both domains. The threshold was determined using a run time analysis, explained in Section 4.3. Table 1 shows transformations generated by our system for a set of some regular responses. As can be seen from the examples, our transformation approach adds interesting domain styles to the responses. Also, there is a clear distinction between the transformations generated from the two domains. For example, the responses transformed using politics domain are very formal, with usages of phrases that can be associated with politics, such as *I'll lead*, *has announced*, *officially declared*, etc. In contrast, the responses generated using the entertainment word graph consist of personal opinion expressions and emoticons. As is evident from some of the examples, not all the responses are coherent. We plan to work towards this as part of our future work by introducing other syntactic constraints and performing optimization to improve transformations. Furthermore, we also plan to perform manual evaluation to judge the quality of our responses. Firstly, we will check if human judges can distinguish between responses generated from both the domains. Secondly, we will also ask judges to rate the responses on the basis of the linguistic quality and the extent of factual information that the transformed response retains.

### 4.3 Runtime analysis

Table 2 shows the results of our run-time analysis when generating relevant patterns from the word-graphs. As can be seen from the table, the graph constructed using tweets related to *politics* contains more nodes than the one constructed with tweets from *entertainment*. Political tweets are generally longer and contain formal language. In contrast, entertainment tweets are shorter, consisting of multiple informal patterns (usage of acronyms, interjections, etc.). We experimented with the pruning threshold ($t_{perc}$) by varying its values between 0.3 and 0.6. As a result of pruning edges, searching for patterns between pairs of words becomes significantly faster. As can be seen, there is a notable impact on run-time (lower is better) when the graph is pruned using setting $t_{perc}$ as 0.5. However, further increasing $t_{perc}$ to 0.6 has a minor effect on run time at the cost of removing

| |
|---|
| **Regular response:** The bond market is hot.<br>**Output (Entertainment):**<br>Discover the hottest year bond market is literally incredible hot.<br>Presenting the most incredible bond market is pretty special hot.<br>**Output (Politics):**<br>Lifting the bond market is absolutely amazing hot.<br>I'll lead the bond market is simply amazing hot. |
| **Regular response:** Costco has the best prices.<br>**Output (Entertainment):**<br>Honestly love Costco has started correctly this week's pretty good promo prices ... ;)<br>Awww love Costco has added sparkle this week's pretty good promo prices ... smh.<br>**Output (Politics):**<br>Costco has passed restoring the best apprentice prices.<br>Costco has announced major part of good prices. |
| **Regular response:** He is a loser.<br>**Output (Entertainment):**<br>He is literally making a total loser !!! xoxoxo<br>He is absolutely making a lil real loser !!! yasssss<br>**Output (Politics):**<br>Interesting facts he is busy making a loser.<br>Disturbing facts he is officially declared a loser. |
| **Regular response:** The food is very tasty.<br>**Output (Entertainment):**<br>Reliving the ultimate food is very tasty.<br>Meet the next big food is very tasty.<br>**Output (Politics):**<br>Lets celebrate the greatest national food is very tasty.<br>Learning about the global food is very tasty. |
| **Regular response:** Pool is a nice game.<br>**Output (Entertainment):**<br>#fbf family pool is finally got a twist easy nice bounce back game !!!!! xoxoxo<br>#fbf family pool is finally got a tasting ; nice kick ass game !!!! xoxo.<br>**Output (Politics):**<br>Pool is worth a nice show amazing game !!<br>Pool is not worth a shutdown it's nice day specific game ! :) |

**Table 1.** Response transformation examples.

several paths from the graph. Therefore, to balance the trade-off, we set $t_{perc}$ to 0.5 for both domains.

## 5 Conclusions and Future Work

In this work, we presented an approach to transform regular chatbot responses to the responses which mimic certain domain-specific linguistic styles. We constructed word-graphs using tweets collected from two different domains – politics and entertainment. Our initial results show that the word-graph is fairly able to generate patterns that can transform existing chatbot responses. Furthermore, our graph pruning analysis shows that the patterns are generated pretty fast without compromising on the relevance or quality. The patterns from the two different domains show significant stylistic differences. Future work includes developing a

**Table 2.** Average run-time for pattern generation and effect of pruning the graphs – Entertainment and Politics. (N and E refer to number of nodes and edges, respectively.)

| | **Entertainment** | | | **Politics** | | |
|---|---|---|---|---|---|---|
| **t_perc** | **N** | **E** | **Avg time (in secs.)** | **N** | **E** | **Avg time (in secs.)** |
| 0.0 | 271427 | 697059 | 3.05 | 277127 | 836056 | 3.01 |
| 0.3 | 22273 | 187734 | 0.62 | 25830 | 272962 | 0.76 |
| 0.4 | 22273 | 160917 | 0.36 | 25830 | 233974 | 0.51 |
| 0.5 | 22273 | 133857 | 0.05 | 25830 | 194954 | 0.05 |
| 0.6 | 22273 | 107262 | 0.03 | 25830 | 155979 | 0.03 |

comprehensive model that can include multiple constraints and decide the best possible transformed response in a given scenario, developing rules to improve grammatical and syntactic correctness of the generated responses.

## References

1. Bamman, D., O'Connor, B., Smith, N.A.: Learning latent personas of film characters. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL). p. 352 (2014)
2. Bird, S.: Nltk: the natural language toolkit. In: Proceedings of the COLING/ACL on Interactive presentation sessions. pp. 69–72. Association for Computational Linguistics (2006)
3. Bruijnes, M., op den Akker, R., Hartholt, A., Heylen, D.: Virtual suspect william. In: International Conference on Intelligent Virtual Agents. pp. 67–76. Springer (2015)
4. Duplessis, D., Letard, V., Ligozat, A.L., Rosset, S.: Joker chatterbot re-wochat 2016-shared task chatbot description report. In: RE-WOCHAT: Workshop on Collecting and Generating Resources for Chatbots and Conversational Agents-Development and Evaluation Workshop Programme (May 28 th, 2016). p. 45
5. Fialho, P., Coheur, L., Curto, S., Cláudio, P., Costa, Â., Abad, A., Meinedo, H., Trancoso, I.: Meet edgar, a tutoring agent at monserrate. In: ACL (Conference System Demonstrations). pp. 61–66. Citeseer (2013)
6. Filippova, K.: Multi-sentence compression: finding shortest paths in word graphs. In: Proceedings of the 23rd International Conference on Computational Linguistics. pp. 322–330. Association for Computational Linguistics (2010)
7. Kubon, D., Hladk, B.: Politician re-wochat 2016 - shared task chatbot description report. In: RE-WOCHAT: Workshop on Collecting and Generating Resources for Chatbots and Conversational Agents-Development and Evaluation Workshop Programme (May 28 th, 2016). p. 43
8. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: Proceedings of The 31st International Conference on Machine Learning. pp. 1188–1196 (2014)
9. Li, J., Galley, M., Brockett, C., Gao, J., Dolan, B.: A persona-based neural conversation model. arXiv preprint arXiv:1603.06155 (2016)
10. McCrae, R.R., Costa Jr, P.T.: A five-factor theory of personality. Handbook of personality: Theory and research 2, 139–153 (1999)
11. Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N., Smith, N.A.: Improved part-of-speech tagging for online conversational text with word clusters. Association for Computational Linguistics (2013)
12. Řehůřek, R., Sojka, P.: Software Framework for Topic Modelling with Large Corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. pp. 45–50. ELRA, Valletta, Malta (May 2010), `http://is.muni.cz/publication/884893/en`
13. Rushforth, M., Gandhe, S., Artstein, R., Roque, A., Ali, S., Whitman, N., Traum, D.: Varying personality in spoken dialogue with a virtual human. In: International Workshop on Intelligent Virtual Agents. pp. 541–542. Springer (2009)