A Unified Framework for Pun Generation with Humor Principles

EMNLP 2022

UCLA

Yufei Tian[1], Divyanshu Sheth[2], and Nanyun (Violet) Peng[1]

[1]University of California Los Angeles, [2]Indian Institute of Technology Kharagpur

PlusLab

1. Introduction

Homographic Pun: words that are written the same

*I used to be a banker but I lost **interest**.*

Homophonic Pun: words that sound alike

Pun word: ***dyed***, Alternative word: ***died***

*Yesterday I accidentally swallowed some food coloring. The doctor says I'm OK, but I feel I've **dyed** a little inside.*

Motivation:

A. Recent success in computational humor theories [1][2] identify three important linguistic traits of puns *(ambiguity, distinctiveness, and surprise),* and show high correlations with human judgments

B. How to incorporate the above attributes to generate better puns is still an open problem [2] integrates surprise, [3] integrates ambiguity only

C. There lacks a unified generation framework for both types of puns.

Our proposed method:

- A theoretically-motivated approach to *incorporate all three linguistic attributes* of puns to pre-trained LMs.
- A *unified framework* to generate both pun types by converting homographic puns to homophonic ones.
- We hypothesize that *there is a learnable structure for puns.*
- We first compose a context word and a phrase from non-pun corpus, and then use a pun label predictor to steer the GPT-2 model to produce puns at inference time.

Pun pair: Principal-principle

-	D2	D1	-	A	A
The	head	**teacher**	exercises	him	occasionally,
D2				A	D1
as the guiding *principal* of				their	school.

Pun pair: Taxi-tax

-	D2	D1	D1	A
An	unlucky	cab	**driver**	might
D2		A	-	A	A
get in all that *taxi* trouble		by	forgetting	to	tell
A	D1	A	-	A	D1
his	passenger	how	much	he	wanted.

A	Ambiguous
D1	Distinct to the first meaning (pw)
D2	Distinct to the second meaning (aw)

Figure 1: Two random examples generated by our model, and the predicted labels. The context words and the extracted phrases are in boldface.

2. Methodology

As shown in Figure 2, our framework consists of three parts.

Step 1. Given the pun word pair, we extract from a non-pun corpus

i) **a context word** that supports the meaning of the pun word, and

ii) **a phrase** that is both characteristic to the alternative word (aw) and compatible with the pun word (pw).

Step 2. Next, we train **a discriminator** on existing homophonic pun data to learn the structure of a pun – the type of each word in the sentence.

Step 3. At inference time, a label predictor is used **re-rank** the tokens generated by the base language model, GPT-2.

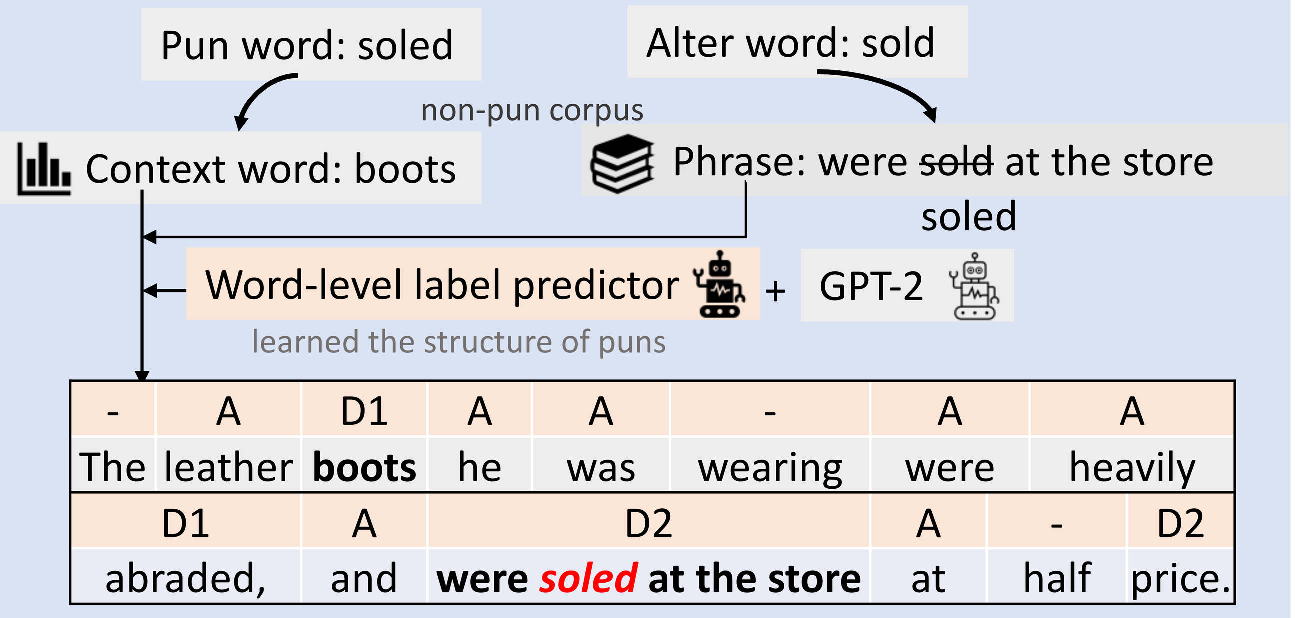

Figure 2: An illustration of our approach. The pw-aw pair (e.g., *taxi-tax*) is the input, and the target output is a pun sentence. After carefully extracting a context word and a phrase, we train a word-level label predictor to steer the LM's generation. A '-' means the label predictor is less confident and thus we do not intervene the generation process.

Obtaining the Context Words

Goal: look for a context word that is distinct to pw.

Method: Retrieve sentences from a non-pun corpus containing pw. Extract keywords using RAKE. Take the words that uniquely co-occur with the target pun word based on the TF-IDF values.

Retrieving the Characteristic Phrase

Goal: look for an ideal phrase that contains aw and replace it with pw to arouse surprise.

Method: To be characteristic of pw and compatible with aw, run RoBERTa for mask in-filling to obtain the probability of words in the masked position. See Table 1 for a more concrete example.

Retrieved Phrase	Tax	Taxi
get in all that <mask> trouble	✓	✓
an export <mask> on agricultural products	✓	×
a new <mask> was created	✓	×
made <mask> deductible against income	✓	✓

Table 1: An example of the retrieved phrases which are characteristic of the alternative word, 'tax'. The pun-pair is 'tax-taxi'. A '✓' means the phrase is compatible with the corresponding word according to our mask in-filling method.

Discriminative Decoding

At each step, we get the predicted type of next token, L, and then re-rank the candidate next words by the corresponding label. if our label predictor is less confident, we do not intervene in the LM's generation.

Algorithm 1 Discriminative Generation (Single Step)

```
function DISCRIMINATIVEGEN
Parameters: pun word (pw), alter word (aw), predicted
  next word label L, confidence c and its threshold T
cands = gpt2.gen_next_word(num = N)
  if c > T then
    if L==A then sort(cands, pw+aw)
    if L==D1 then sort(cands, pw)
    if L==D2 then sort(cands, aw)
  ▷ Sort according to semantic relevance in Section 3.2
return cands
```

3. Label Predictor

Challenge One major challenge for Step 2 is that there are no ground truth labels. To this end, we collect a small amount of human annotations and boost from weak, unsupervised models to stronger, supervised models.

Dataset SemEval 2017 Task 7 (Miller et al., 2017) 1500 human written pun sentences

Ground Truth Label Curation

Unsupervised: Use glove embedding for semantic similarity For a predefined threshold T and every word *tw* in the sentence, if |cos(tw, pw) – cos(tw, aw)| > T, label *tw* as D1/D2. Otherwise, label as A. Achieved 72.9% label accuracy.

Supervised: Finetune a BERT model for classification. Provide this BERT classifier with *less noisy data* so that it can learn the task better than the unsupervised approach. Achieved 84.6% label accuracy.

Training

We train another BERT classifier. Recall that we need to predict the *next token type*, different from the label curation classifier.

Category	A	D1	D2
$\mathbf{Bert}_n$	0.81	0.68	0.60
- human labeled train data	- 0.02	- 0.06	- 0.05
+ high confidence (T=0.9)	+0.03	+0.02	+0.05

Table 2: The F1 scores of $\mathbf{Bert}_n$ and its ablations on human annotated testset.

4. Migrate to Homographic Puns

Convert the task of generating homographic puns to homophonic puns by leveraging a reverse dictionary (RD) and a word-sense-disambiguation (WSD) model.

Example:

Target pun word: *sentence*

Meaning 1: a set of words…

Meaning 2: the conviction…

clause

punishment

Pun pair	sentence ⟹clause-punishment
Pun-GAN	Due to the sentence it is in the United States.
AmbiPun	The sentence is ungrammatical. The jury didn't hear it.
Ours	The language on a two-page sentence for fraud is full of guilt.
Human	The judge has got a stutter. Looks like I am not getting a sentence.

5 Ablation

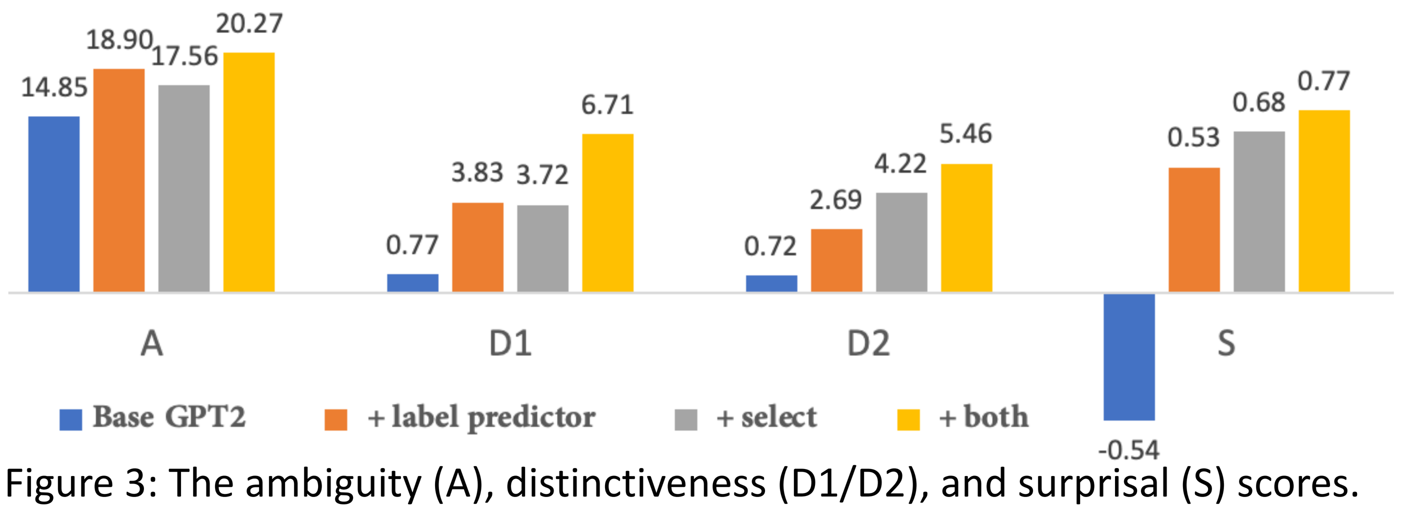

Figure 3: The ambiguity (A), distinctiveness (D1/D2), and surprisal (S) scores. Bar chart showing the improvement after adding each component.

6 Results

Homophonic Pun	Success	Informative	Funny
LCR	39%	2.11	2.14
SurGen	42%	2.74	2.35
Base GPT-2	16%	2.52	1.56
+ label predictor*	40%	2.96	2.04
+ select*	49%	3.35	2.57
+ both*	56%	3.60	2.96
Human	**89%**	**4.56**	**4.04**

Homographic Pun	Success	Informative	Funny
Pun-GAN	20%	1.72	1.54
AmbiPun	44%	2.76	2.40
Base GPT-2	14%	2.17	1.55
+ label predictor*	29%	2.84	2.03
+ select*	43%	3.13	2.51
+ both*	47%	3.32	2.83
Human	**85%**	**4.23**	**3.87**

Human Evaluation:

1. Pun Success Rate
2. Informative (or specific, to avoid general outputs)
3. Funniness Level

UCLA

SCAN ME