# Mind-blowing Compound Words:

## Identifying whether generated compound words are real words

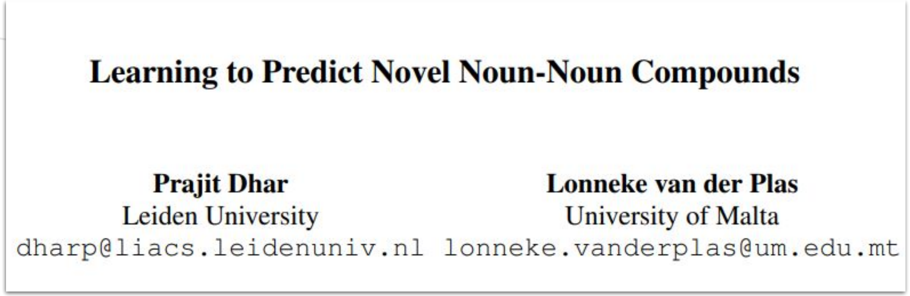
# Hello!

| | |
|---|---|
| **Timothy Cruz** | *Data Science* |
| **Joseph Park** | *Data Science & MCB* |
| **Danny Siu** | *Data Science* |

# Question:

How well can a model recognize whether generated compound words are real words or not?

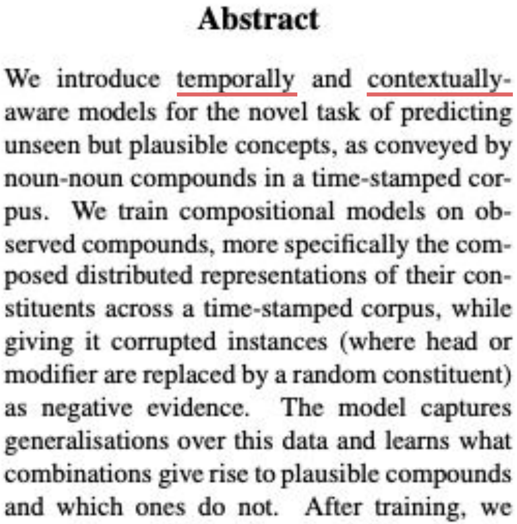
# Inspiration



Learning to Predict Novel Noun-Noun Compounds

Prajit Dhar
Leiden University
dharp@liacs.leidenuniv.nl

Lonneke van der Plas
University of Malta
lonneke.vanderplas@um.edu.mt

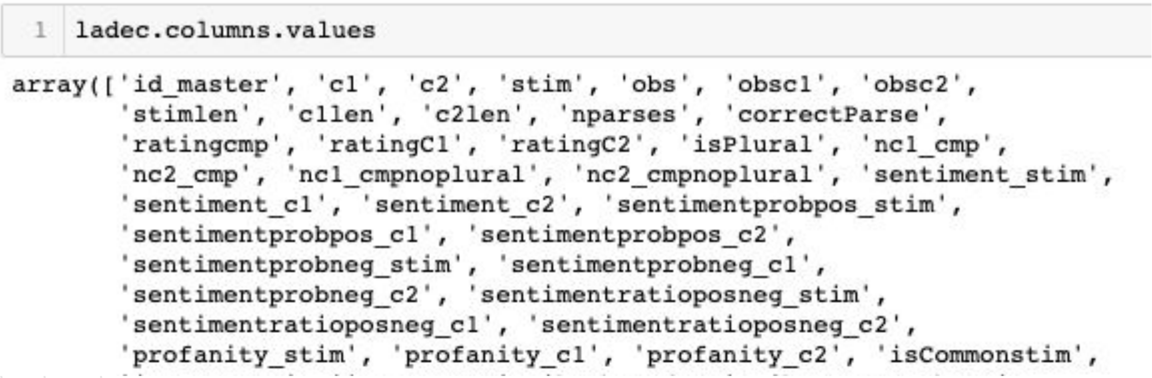


Abstract

We introduce temporally and contextually-aware models for the novel task of predicting unseen but plausible concepts, as conveyed by noun-noun compounds in a time-stamped corpus. We train compositional models on observed compounds, more specifically the composed distributed representations of their constituents across a time-stamped corpus, while giving it corrupted instances (where head or modifier are replaced by a random constituent) as negative evidence. The model captures generalisations over this data and learns what combinations give rise to plausible compounds and which ones do not. After training, we

# LADEC

- Large Database of English Compounds
- Nearly 9000 English compound words
- 84 features



```
ladec.columns.values
```

```
array(['id_master', 'c1', 'c2', 'stim', 'obs', 'obsc1', 'obsc2',
       'stimlen', 'c1len', 'c2len', 'nparses', 'correctParse',
       'ratingcmp', 'ratingC1', 'ratingC2', 'isPlural', 'nc1_cmp',
       'nc2_cmp', 'nc1_cmpnoplural', 'nc2_cmpnoplural', 'sentiment_stim',
       'sentiment_c1', 'sentiment_c2', 'sentimentprobpos_stim',
       'sentimentprobpos_c1', 'sentimentprobpos_c2',
       'sentimentprobneg_stim', 'sentimentprobneg_c1',
       'sentimentprobneg_c2', 'sentimentratioposneg_stim',
       'sentimentratioposneg_c1', 'sentimentratioposneg_c2',
       'profanity_stim', 'profanity_c1', 'profanity_c2', 'isCommonstim',
```

# Methodology

1. Gather compound words from LADEC
2. Use LADEC word constituents to form corrupt words
3. Split dataset into validation, training, and test data
4. Train model on training dataset
5. Tune model using validation dataset
6. Test model on test dataset
7. Analyze results
8. Make conclusions

# Dataset

**ELP: English Lexicon Project, based in the US**

- 3149 compound words

- Features: “C1”, “C2”, “Stim”, “isCommonStim”

# A Simple Example:

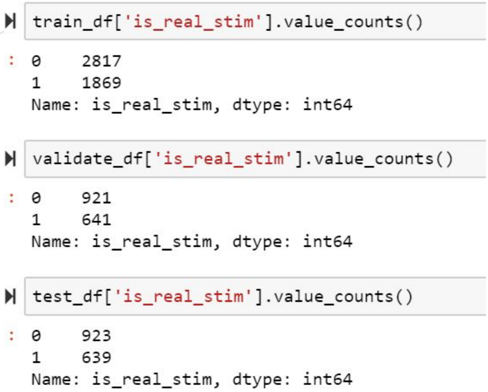
- “After” + “Math” = “Aftermath” ✓
- “Gold” + “Fish” = “Goldfish” ✓
- “After” + “Fish” = “Afterfish” ✗
- “Gold” + “Math” = “Goldmath” ✗

# Final Dataset

- 7812 total compound words
  - 3149 valid compound words
  - 4663 “corrupt” compound words
- 2339 Features
  - 1168 unique c1 words
  - 1170 unique c2 words
  - isCommonStim

# Train, Test, & Validation

- Split our data set into 60 / 20 / 20



```
train_df['is_real_stim'].value_counts()
```

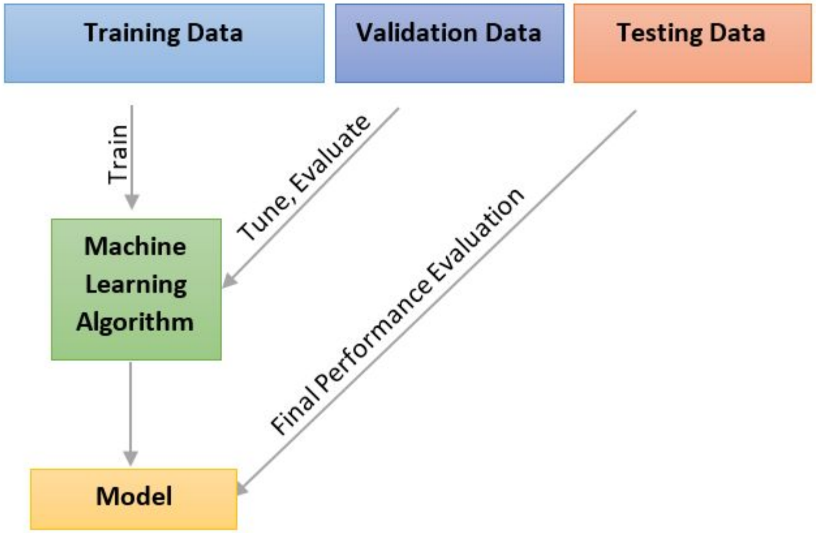
```
0    2817
1    1869
Name: is_real_stim, dtype: int64
```

```
validate_df['is_real_stim'].value_counts()
```

```
0    921
1    641
Name: is_real_stim, dtype: int64
```

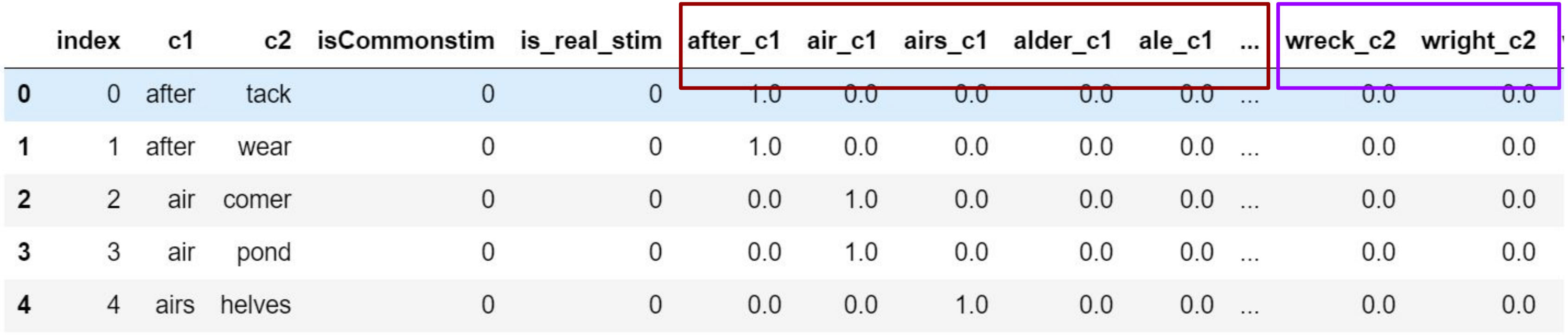
```
test_df['is_real_stim'].value_counts()
```

```
0    923
1    639
Name: is_real_stim, dtype: int64
```



# Training our Model

- We trained our model on **4686** compound words
- Used one-hot encoding of constituent words as features

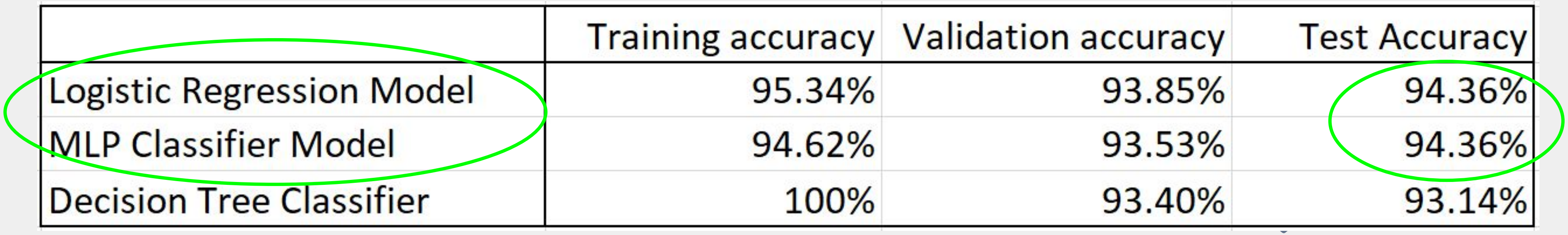


| | index | c1 | c2 | isCommonstim | is_real_stim | after_c1 | air_c1 | airs_c1 | alder_c1 | ale_c1 | ... | wreck_c2 | wright_c2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | after | tack | 0 | 0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 |
| **1** | 1 | after | wear | 0 | 0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 |
| **2** | 2 | air | comer | 0 | 0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 |
| **3** | 3 | air | pond | 0 | 0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 |
| **4** | 4 | airs | helves | 0 | 0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 |

# Evaluating the best model

- Created a Logistic Regression Model, Multi-Layer Perceptron Model, and Decision Tree Classifier Model.
- Used validation accuracy to gauge quality of model and for hyperparameter tuning
- Used test accuracy to test our model on real unseen data

# Results



| | Training accuracy | Validation accuracy | Test Accuracy |
|---|---|---|---|
| Logistic Regression Model | 95.34% | 93.85% | 94.36% |
| MLP Classifier Model | 94.62% | 93.53% | 94.36% |
| Decision Tree Classifier | 100% | 93.40% | 93.14% |

**Conclusion:** We are able to distinguish between “real” and “corrupt” compound words with fairly high accuracy, by just using the constituents as features.

# Future Exploration…

- Variables to consider:
    - Commonality of certain constituents
    - Constituent words’ synonyms
    - Parts of speech & corresponding location within word
    - Other types of compound words
- Experimenting with different hyperparameters and models

# Real-World Applications of Our Project

- Teach us about how we, as a society, decide on compound words
  - Importance of features: frequency of constituents, synonyms of existing constituents, POS of constituents, etc.
- Predicting the existence of compound words could have linguistic implications
  - English word formation, semantics, etc.

# Works Cited

- Inspiration Research paper: https://www.aclweb.org/anthology/W19-5105.pdf
- LADEC Research article: https://www.ncbi.nlm.nih.gov/pubmed/31347038 https://link.springer.com/article/10.3758/s13428-019-01282-6#Sec2

Thank you & good luck on finals!