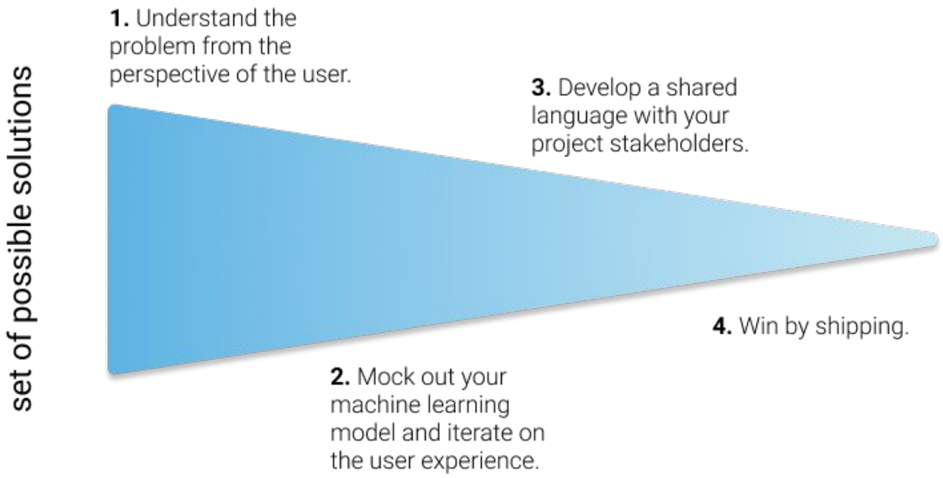
Week 6: Supervised and unsupervised applications

Text Analytics and Natural Language Processing

Instructor: Benjamin Batorsky

Scoping an NLP project

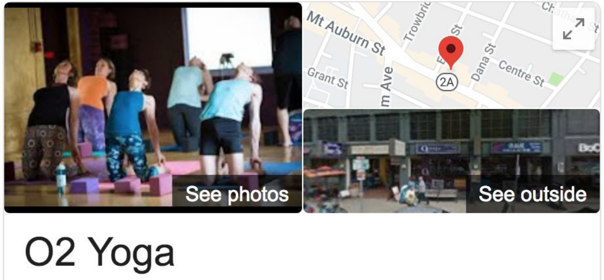
- Setup
  - Understand the problem
  - Inventory of solutions
    - Impact
    - Feasibility
    - Requirements
  - Setting up code base
- Data collection/labelling/sourcing
- Model exploration
- Deployment
- (throughout) Debugging and testing



https://www.jeremyjordan.me/ml-requirements/

Unsupervised use-case: Customer segmentation

- Small/Mid-sized businesses that straddle multiple categories
- Customer questions
    - Sales: “Which businesses are similar to this lead?”
    - Marketing: “How do we better personalize ad campaign messaging?”
    - Guidance: “How do we surface the most relevant recommendations to customers?”



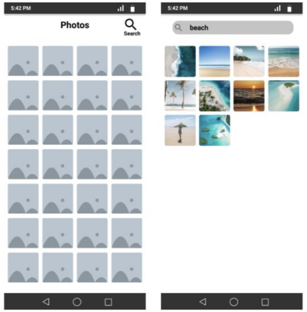
“...offers classes 7 days a week. Our vegan cafe opened in July of 2013... We also have a retail store selling a limited selection of US-made yoga gear...peruse the retail, enjoy the cafe, or get a massage with one of the body workers in the Wellness Center…”

Yoga studio, cafe AND retail?!

Understanding the problem

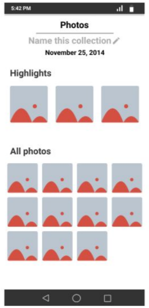
Review filtering mechanism

- Who are the end users?
    - Viewers
    - Creators
    - Internal
- Is this an NLP problem?
    - Are there text features?
    - Are the text features informative?
- What are some possible solutions?



Text query

Topic-centric query



Group photos and select highlights

https://www.jeremyjordan.me/ml-requirements/

Understanding the problem: Customer segmentation

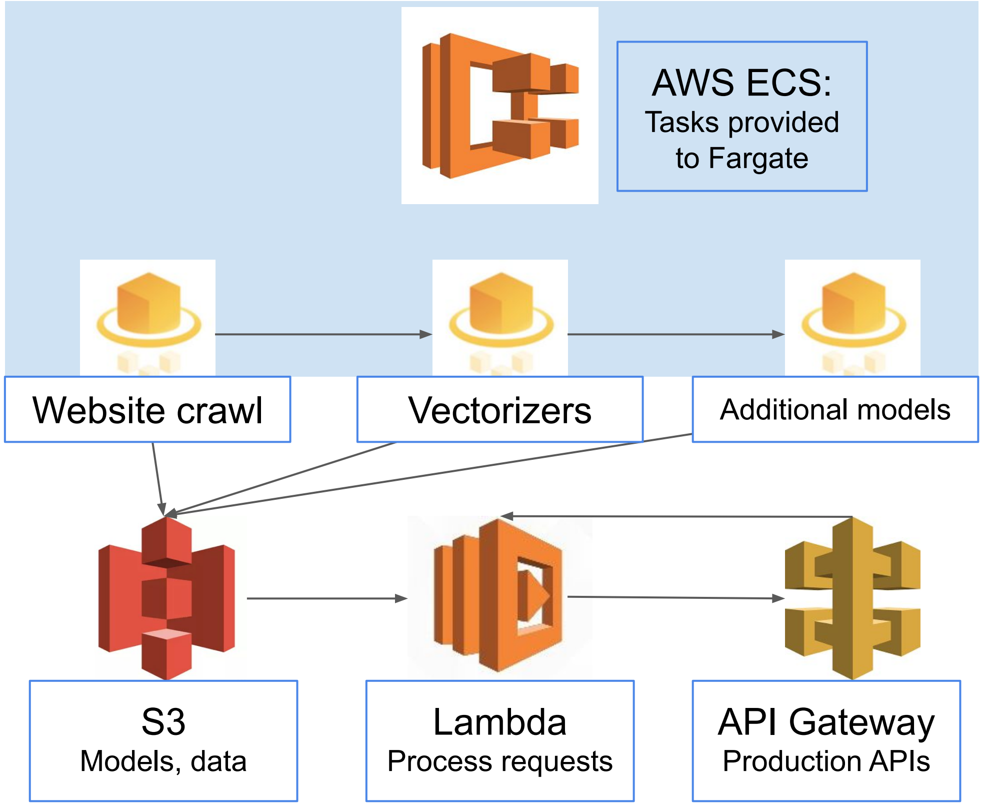
- Who are the end users?

- Is this an NLP problem?
  - Are there text features?
  - Are the text features informative?

- What are some possible solutions?

Implemented solution: Segmentation Pipeline

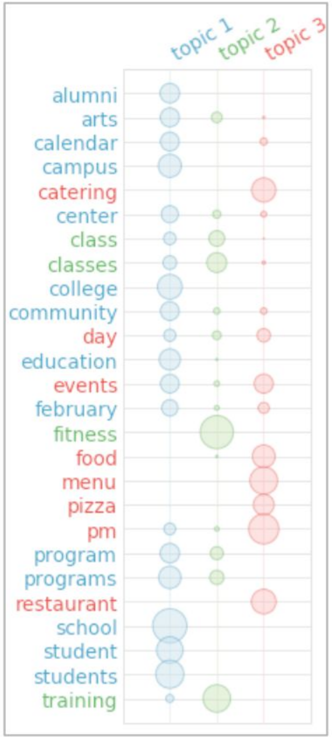
- Data ingestion
  - Daily crawl of new websites
- Preprocessing
  - Stripping punctuation
  - Number/URL tokens
- Vocabulary generation
  - Gensim phrase identification
    - E.g. “ice”, “cream” converted to “ice cream”
  - Vocabulary generated from presence across industry
- Information extraction/vectorization
  - Term Frequency-Inverse Document Frequency

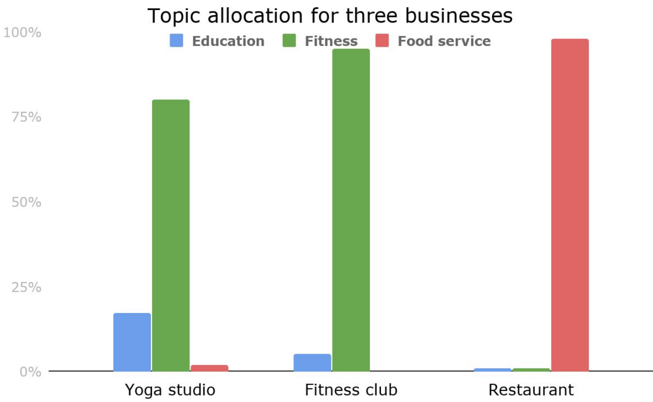


Segmentation Pipeline (continued)

- Topic modelling
  - Non-negative matrix factorization (NMF)
  - Identify interpretable topics and relevant words
- Output
  - TF-IDF + NMF transform on processed website text
  - Calculate cosine similarity across all businesses
  - Identify populations ranked by similarity
- Feedback
  - Feedback from internal users on similarity of results
  - Feedback used to reweight pairwise similarity and train new models

Product similarity

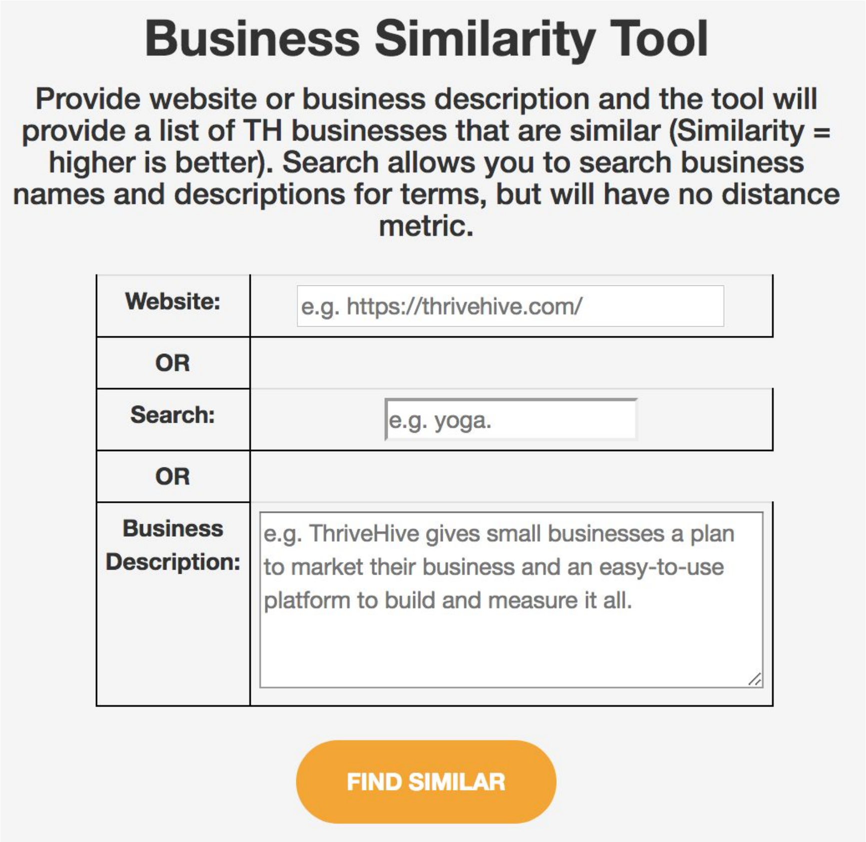




Circles are sized according to "relevance" to each topic

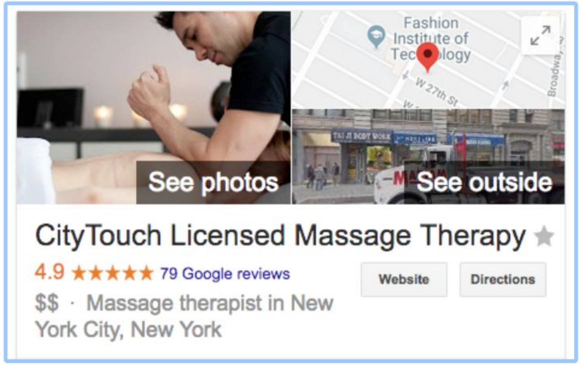
Personalized targeting for marketing/sales

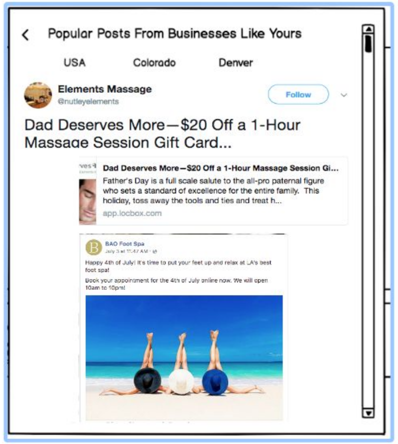
- Interactive web application over NLP API
- Three options
  - Website - On-request text extraction
  - Description - Treated as document
  - Keyword search - Simple regex search
- Similar business plus success metrics
- Sales
  - Regular usage during lead calls/prospecting
  - Source of feedback for similarity
- Marketing
  - Ad-hoc building of targeting populations
  - “Success stories” for similar businesses



Personalized guidance for businesses

- Content generation a major issue for business owners
  - "What do I talk about?"
- Surface top-performing posts from similar businesses
- Initial testing: 80% of content identified as relevant
  - 75% of content identified as relevant claimed to "give ideas for new content"





Implemented solution

Solution	Data source	Metric of interest	Feasibility	Impact
Customer “similarity” score	Website text	Customer feedback (internal/external)	High: Limited modelling required, light-weight process	High: Enables comparing business without relying on categories

Notebook use-case: Recommending similar movies

Goal: Can we recommend similar movies to a particular movie or set of movies?

https://github.com/bpben/nlp_lessons/blob/master/notebooks_instructor/week_6_usecases.ipynb

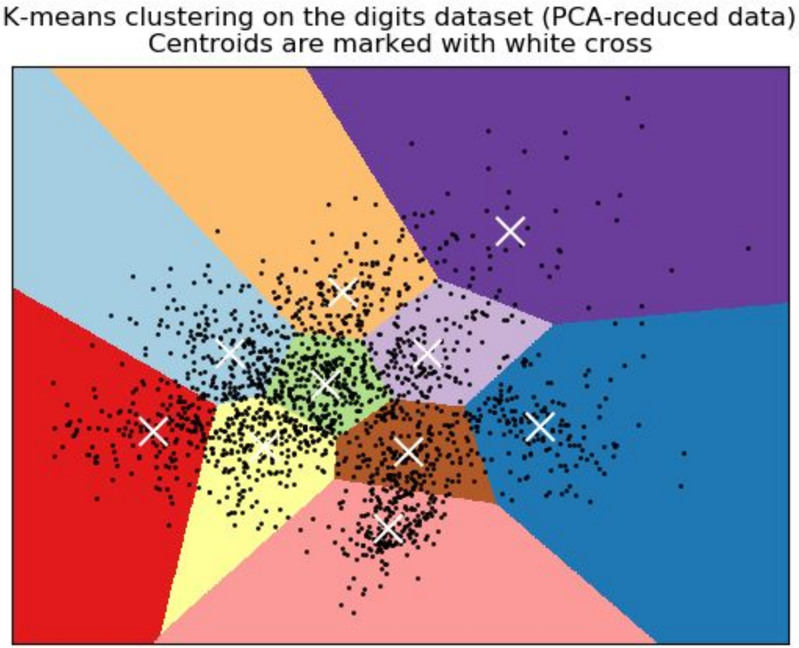
Notebook use-case: Categorizing movies

Goal: Can we create informative groupings of movies?

https://github.com/bpben/nlp_lessons/blob/master/notebooks_instructor/week_6_usecases.ipynb

K-means clustering

- Clustering: Unsupervised approach
  - “Identify patterns in data”
- K-means
  - Initialize K center points
  - “Draw” segment around center point
  - New center = center of the segment
  - Repeat until no significant change
- Optimizes for “inertia”
  - Sum of squared distances from center of cluster



https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_digits.html

Notebook use-case: Identifying actors/locations

Goal: Can we extract the actors and locations mentioned in reviews?

https://github.com/bpben/nlp_lessons/blob/master/notebooks_instructor/week_6_usecases.ipynb

Named-Entities

- Named-entity: A real-world named object (e.g. person, place, organization)
    - New York City is different than just an assembly of three words “new”, “york” and “city”
- To identify these
    - Dictionaries
    - Pattern-matching
    - Models

In June 2020 [DATE], I took a course at Harvard Extension School [ORG] in Cambridge [LOC].

Why is NER difficult?

- Ambiguity about entity boundaries
- Good performance typically requires a lot of quality labelled data
  - CoNLL 2003 task: ~21k labelled sentences
  - OntoNotes 5.0: 1.4 M articles
- Task complexity
  - Each token needs one tag and one entity type, decision between X tags * Y types
- Performance evaluation
  - What if the method achieves partial match?
  - Are we interested in performance on new data? Out-of-inventory entities?

Harvard [U-ORG (or B-ORG)]

Extension [B-ORG (or I-ORG)]

School [L-ORG]

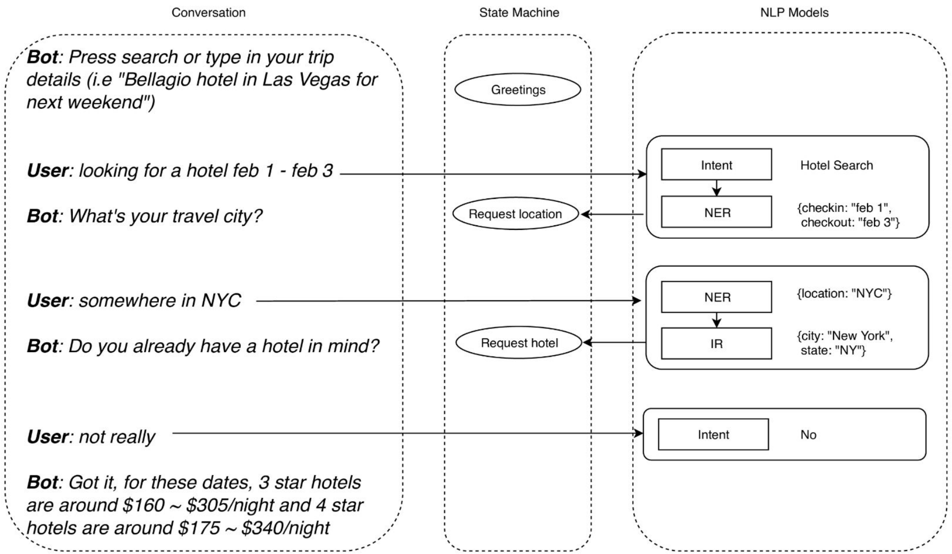
in [O]

Cambridge [U-LOC]

CoNLL 2003: https://www.aclweb.org/anthology/W03-0419/
OntoNotes 5.0: https://catalog.ldc.upenn.edu/LDC2013T19

Why is NER important?

- Named-entities are fundamentally different from other tokens
    - “New York City” != “new” + “york” + “city”
- Having a “complete” inventory is extremely rare
- NER is part of many Machine Learning applications
    - Content recommendation/Search
    - Chatbots
    - Translation



https://www.groundai.com/project/real-world-conversational-ai-for-hotel-bookings/1

NER use-case

- Dataset: Full-text for 22k court cases
- Current NER approach:
  - Regular expressions using inventories
    - Inventories likely to be incomplete and spellings/mentions likely to vary
  - Manual review and hand-labelling
    - Extremely time-consuming
    - Needs to be repeated for new data/new questions
- Proposed approach:
  - Model-based NER
  - Compare versus current approach
    - Performance on new data
    - Performance on entities out-of-inventory

What are the regulatory agencies involved in food safety enforcement?

"China FDA brought a suit against…"

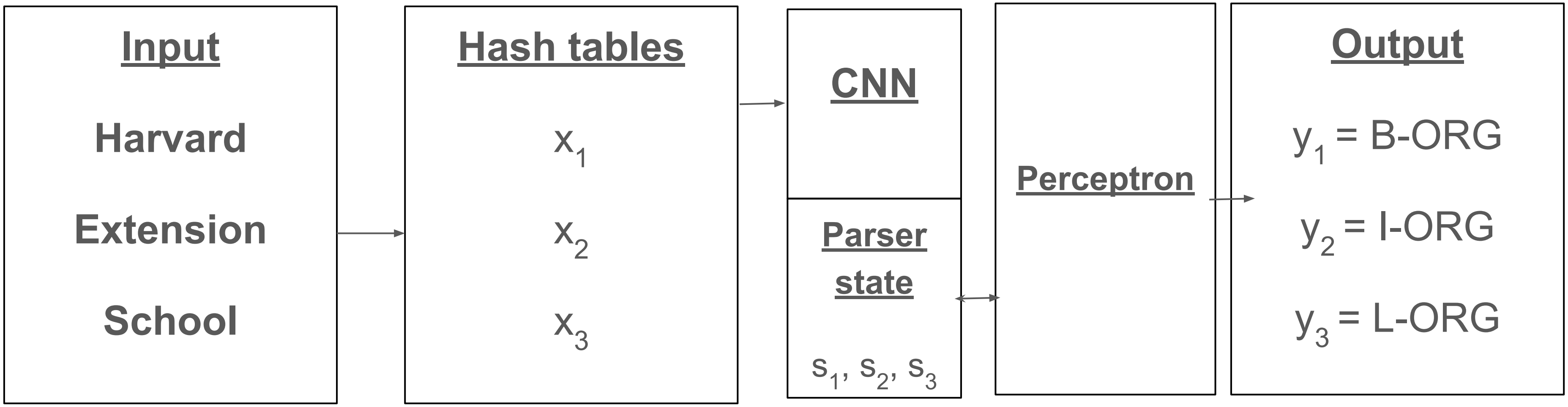
What types of products do they oversee?

"...for selling tainted pork products…"

What is their jurisdiction?

"...in Hangzhou Province."

SpaCy’s NER model



Matthew Honnibal on spaCy’s NER model: https://spacy.io/universe/project/video-spacys-ner-model

Our data and inventories

- Dataset
  - Full-text for 22k court cases prosecuted by government agencies
- Inventories
  - Products
    - Sourced from our database of food inspection results
  - Agencies
    - Curated as part of the previous manual process
  - Locations
    - A selection of cities and all prefectures and provinces in China

被告人丁某某，女，1965年3月13日出生，汉族。因涉嫌犯销售不符合安全标准的食品罪于2014年9月29日被 郑州市 LOCATIONS 公安局须水分局刑事拘留，于同年10月6日被 郑州市 LOCATIONS 公安局须水分局取保候审，于同年10月22日被 郑州市 LOCATIONS 郑州市 LOCATIONS 中原区人民检察院取保候审，经本院决定于同年10月28日被取保候审。

郑州市 LOCATIONS 中原区人民检察院以郑中检公诉刑诉（2014）333号起诉书指控被告人丁某某犯销售不符合安全标准的食品罪，于2014年10月28日向本院提起公诉。本院依法适用简易程序，实行独任审判，公开开庭审理了本案。 郑州市 LOCATIONS 中原区人民检察院指派代理检察员付婧文出庭支持公诉，被告人丁某某到庭参加诉讼。现已审理终结。

郑州市 LOCATIONS 中原区人民检察院指控:2014年9月下旬，被告人丁某某在 郑州市 LOCATIONS 二七区金海市场一男子处低价购进 食盐 PRODUCTS ，在 郑州市 LOCATIONS 中原区伊河路菜市场其所经营的干菜店里予以销售，2014年9月29日， 郑州市盐业管理局 AGENCIES 工作人员在被告人丁某某经营的干菜店内查获“卫群”牌 食盐 PRODUCTS 69袋，27.6公斤。经 河南省盐产品质量监督检验中心 AGENCIES 检验，该盐氯化钠含量达到精制工业盐标准，不含碘。

Using SpaCy’s PhraseMatcher

- PhraseMatcher
  - Given a pattern, extracts matches and can pass to callback function
    - (Match id, start token, end token)
- Custom callback
  - Default entity attribute can’t handle overlap
  - Used custom attribute for Doc objects
    - Doc._.entities
  - For overlapping entities
    - If different types, choose higher priority one
    - If same types, go with longer entity

```
# intialize model
nlp = Chinese()
# initialize the matcher with model vocab
matcher = PhraseMatcher(nlp.vocab)
# add entity inventory as Doc objects from model
or i, c in enumerate(ENTITY_TYPES):
    matcher.add(c, add_entity, *parsed_ents[i])
```

Callback function

Constructing training and test datasets

- Year-split
  - How well will the model perform on future court case data?
  - Train on cases before 2017, test on after
  - Baseline: Inventory with only entities before 2017
- Excluded entities
  - How well does the model identify entities it hasn't seen?
  - Train on cases with 30% of entities from each inventory removed
  - Baseline: Inventory with remaining 70%

Method	Train		Test	
	Docs	Unique entities	Docs	Unique entities
Year-split	13,501	6,707	7,722	4,378
Excluded entities	14,859	8,548	6,364	2,564

Training the model

- Available hyperparameters
  - Dropout
  - Batch size
  - Optimizer
- Outputs loss with each epoch
  - Based on the model predicted tags (e.g. entity/non-entity)*
- Important to note
  - Plateauing: I haven't seen much movement in loss after 15-20 epochs
  - If updating: Not enough to just put new examples, model likely to forget what it's learned

```
nlp_model = Chinese()
ner = nlp_model.create_pipe('ner')
nlp_model.add_pipe(ner)
for l in labels:
    ner.add_label(l)
optimizer = nlp_model.begin_training()
sizes = compounding(1.0, 4.0, 1.001)
epoch = 15
for itn in range(epoch):
    random.shuffle(train_data)
    batches = minibatch(train_data, size=sizes)
    for batch in batches:
        texts, annotations = zip(*batch)
        nlp_model.update(texts, annotations, sgd=optimizer, drop=0.35, losses=losses)
    print("Losses", losses)
```

Losses {'ner': 169404.9403350675}

Losses {'ner': 79686.03164099755}

*More details on NER model loss: https://github.com/explosion/spaCy/issues/3360

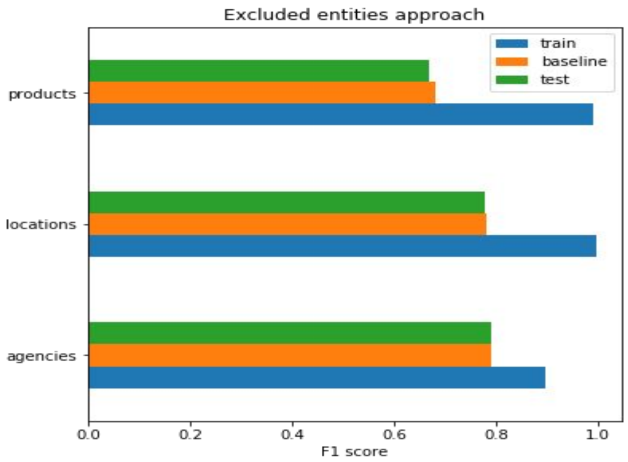
Scoring the result

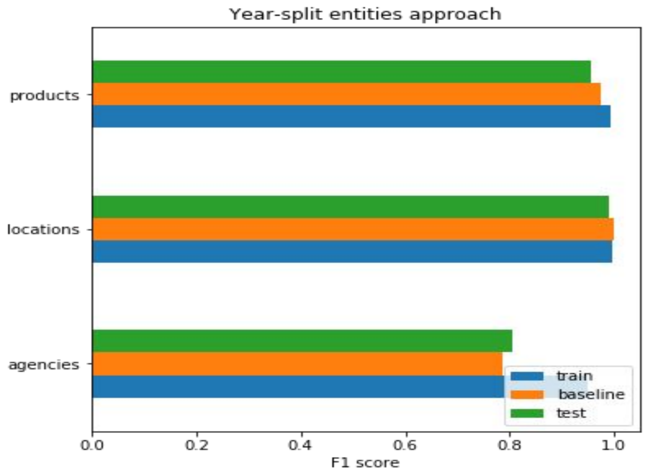
- Built-in SpaCy Scorer
  - Compare model parsing result to "gold standard"
  - Provides entity-level precision ("p"), recall ("r") and F1-score ("f")
  - Adapted to get baseline performance
- How does this model perform on data it has seen?
  - F1-score on training data
- How does this model perform on unseen data?
  - F1-score on test data
- How well does this model identify entities it hasn't seen?
  - Recall of excluded entities

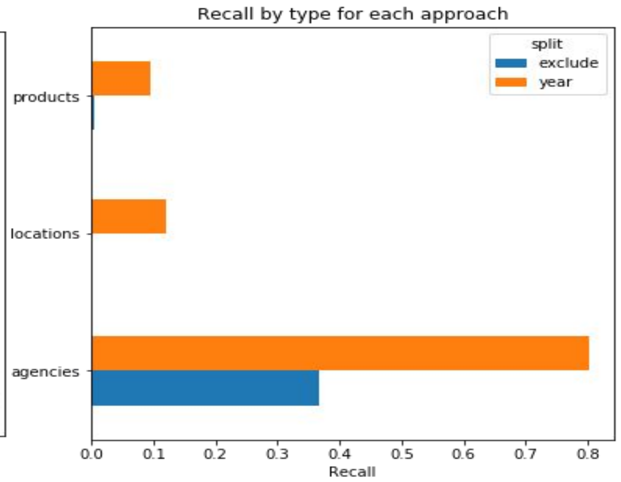
```
scorer = Scorer()
for doc, annot in test_data:
    doc_to_test = full_model(doc)
    gold_text = nlp(doc)
    gold = GoldParse(gold_text, entities=annot.get("entities"))
    scorer.score(doc_to_test, gold)
```

```
{'uas': 0.0,
 'las': 0.0,
 'las_per_type': {'': {'p': 0.0, 'r': 0.0, 'f': 0.0}},
 'ents_p': 97.69166443143628,
 'ents_r': 55.78143651884051,
 'ents_f': 71.0141561506281,
 'ents_per_type': {'agencies': {'p': 88.99794567450354,
   'r': 70.3917674670518,
   'f': 78.60887096774192},
  'locations': {'p': 99.61005302327793,
   'r': 63.799037524366476,
   'f': 77.7805627500673},
  'products': {'p': 97.63231014366795,
   'r': 51.04266349059916,
   'f': 67.03768671561359}},
 'tags_acc': 0.0,
 'token_acc': 100.0,
 'textcat_score': 0.0,
 'textcats_per_cat': {}}
```

Model-based NER has better performance on agencies than other entity types







NER model for identifying actors/locations

https://github.com/bpben/nlp_lessons/blob/master/notebooks_instructor/week_6_usecases.ipynb

Language generation - Cool stuff

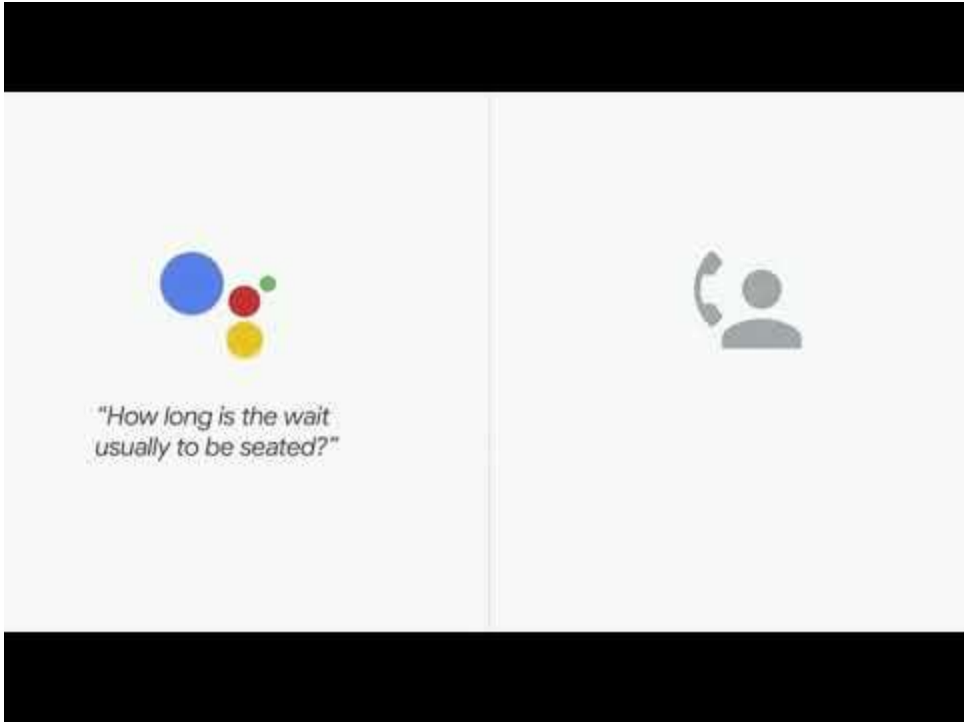
Write with Transformer from HuggingFace:
https://transformer.huggingface.co/doc/distil-gpt2

I am teaching a course at Harvard on **how to develop and learn about language skills, such as German and French.**

Written by Transformer · transformer.huggingface.co

Language generation - useful stuff

Google Assistant making a reservation



Models “learning” language

100 epochs

tyntd-iafhatawiaoihrdemot lytdws e ,tfti, astai f ogoh eoase rrranbyne 'nhthnee e plia tklrgd t o idoe ns,smtt   h ne etie h,hregtrs nigtike,aoaenns lng

700 epochs

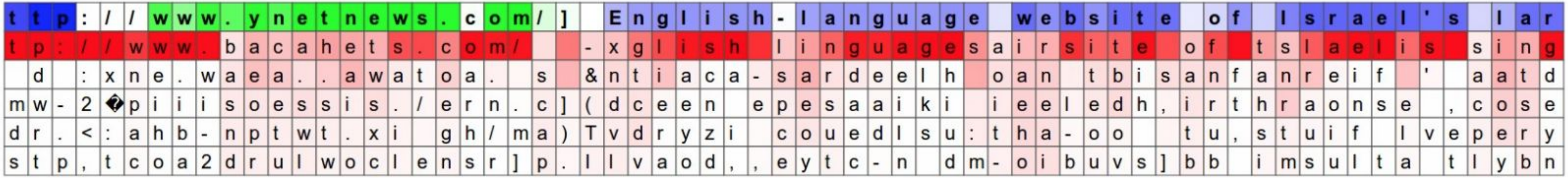
Aftair fall unsuch that the hall for Prince Velzonski's that me of her hearly, and behs to so arwage fiving were to it beloge, pavu say falling misfort how, and Gogition is so overelical and ofter.

2000 epochs

"Why do what that day," replied Natasha, and wishing to himself the fact the princess, Princess Mary was easier, fed in had oftened him. Pierre aking his soul came to the packs and drove up his father-in-law women.

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Neurons as feature learners



desktop_computer

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Chatbot example with Rasa

Natural Language Understanding (NLU)

```
"I am looking for a Mexican restaurant in the center of town"
```

```
{
  "intent": "search_restaurant",
  "entities": {
    "cuisine" : "Mexican",
    "location" : "center"
  }
}
```

Natural Language Generation (NLG)

Template-based

intents:

- greet

responses:

- utter_greet
  - text: “Hello {name}”

Model-based (external to Rasa)

FastAPI call to your NLG model

Adapting our sentiment model for NLG

https://github.com/bpben/nlp_lessons/blob/master/notebooks_instructor/week_6_usecases.ipynb