Compound Data Embeddings:
Handling Text+Graph Data

Akshar Varma*, Northeastern University
Ongoing work with Tanay Mehta and Ravi Sundaram

*Work done while an intern at Amazon

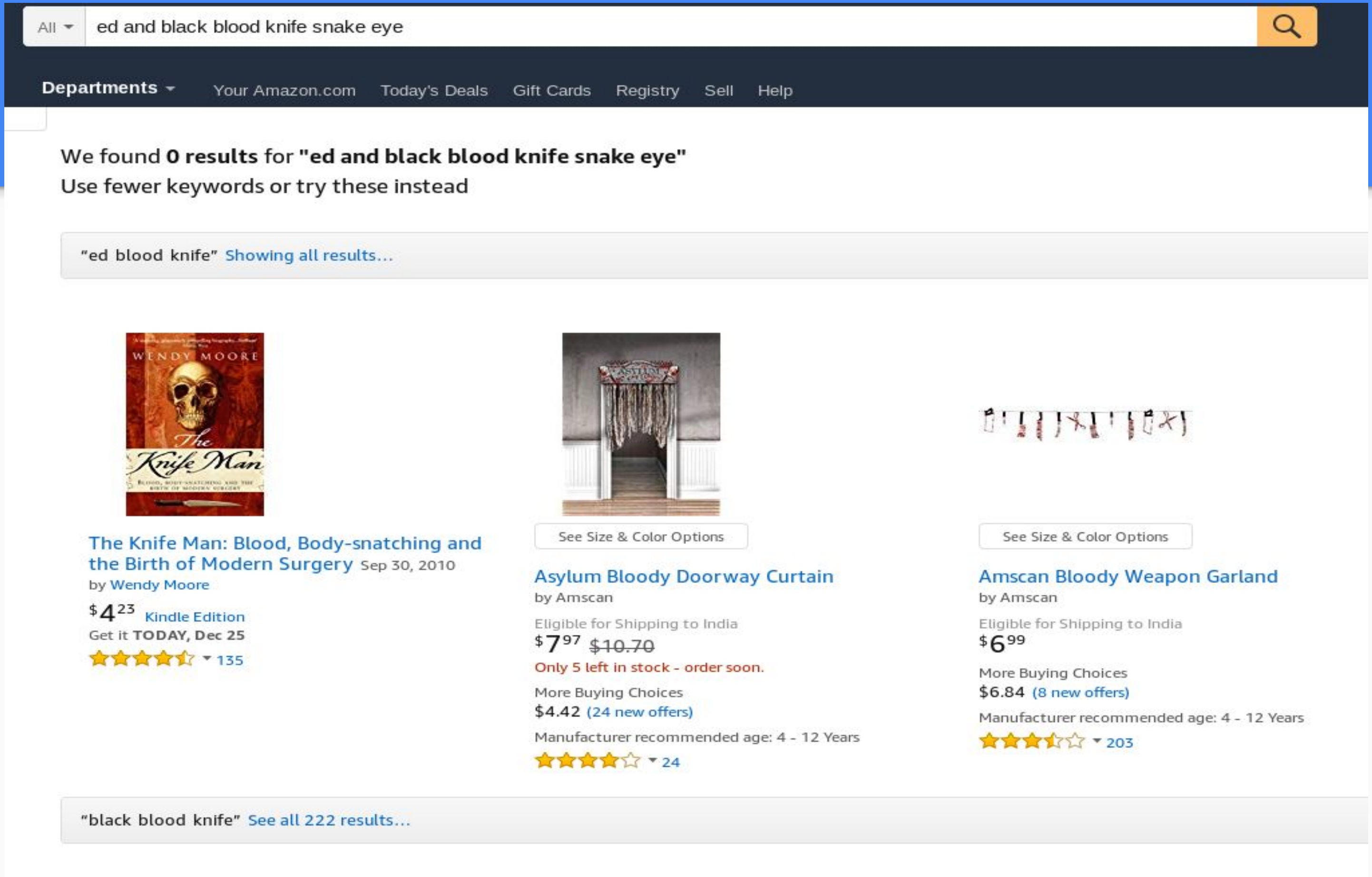
Overview

- A concrete problem
- Our approach and its performance
- General problem
- Theoretical justification

The Problem

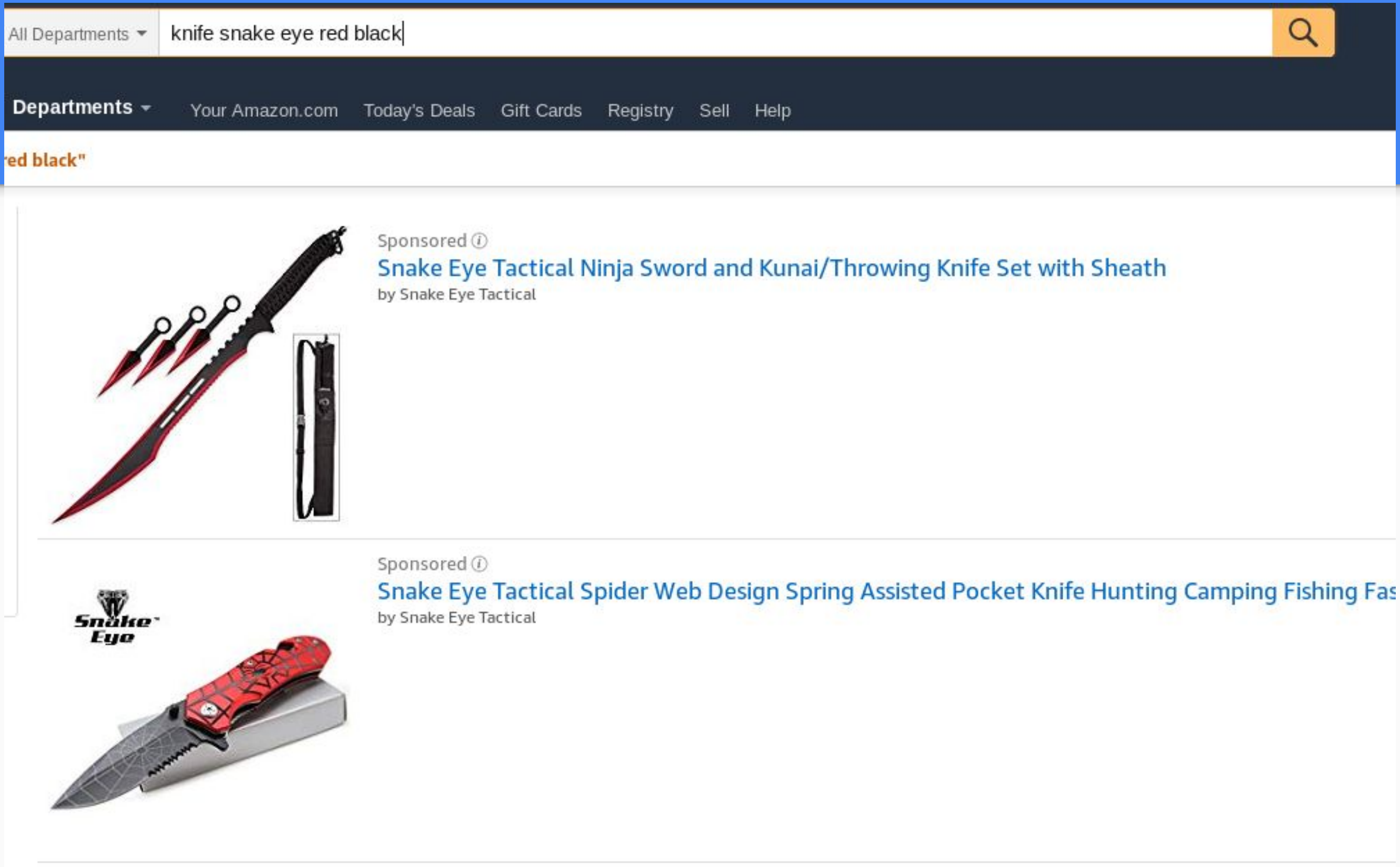
- User makes a (tail) query that yields no products
  - Query is too long/specific
  - Query is too vague
- Existing solution: suggest query reformulations by *dropping random words*.

Dropping random words



Can we do something better?

Example better reformulation



The problem setup

Learn a model that infers related queries given query text

- Training
  - Query-Product behaviour represented as Query-Query graph
    - Query text (as trigrams/words) as node labels
    - Add edge if queries share any product
- Inference
  - Only query text is known.

Get embeddings such that related queries are close in embedding space.

Limitations of existing embedding techniques

- Word/Sentence embeddings:
  E.g. word2vec (Mikolov et al., NIPS 2013), GloVe (Pennington et al., EMNLP 2014), SIF (Arora et al., ICLR 2017), ELMo (Peters et al., NAACL 2018)
  - Cannot incorporate graph data.
- Graph embeddings:
  E.g. node2vec (Grover and Leskovec, KDD 2016), GraphSage (Hamilton et al., NIPS 2017), GAT (Velickovic et al., ICLR 2018)
  - Inference needs full graph.

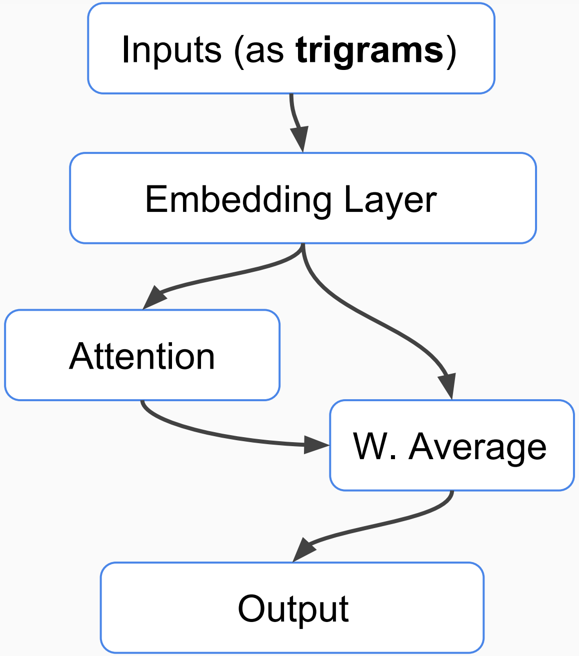
Overview of our approach

- Naive idea:

  Get Graph embeddings, then use an LSTM to map text to the embeddings.
  - LSTMs are very slow.
  - Empirically perform poorly.
- Better idea:
  - Attention network to generate text embeddings.
  - Graph information in loss function (only while training).
  - Recommend *k* queries using KNN search over embeddings.

Attention-based Neural Network

- Attention (*Bahdanau, Cho, Bengio, ICLR 2015*) gives the neural network ability to focus on relevant parts
    - Eg: for "**marble iphone 7 32GB case black**", we focus on '**iphone**', '**7**', and '**case**'.
- Attention Weights for each trigram, using the embeddings of all trigrams.
- Take weighted average of trigram embeddings



Loss Functions

1. **Match Graph embeddings:**
   Use cosine similarity to match node embeddings generated from some graph embedding method.
2. **Positive/Negative Sampling:**
   Maximize similarity wrt +ve samples & minimize similarity wrt -ve samples
   a. Positive samples:
      i. Uniformly random neighbours
      ii. Co-occuring nodes on long random walks
   b. Negative samples are uniformly random non-adjacent nodes

Metrics

- How to measure relevance of recommended queries?
  - Relevant reformulations are not always adjacent on the QQ-graph. Noisy dataset.
  - Ideal measure: Use human-labelled relevance of recommendations
- Measure a weak signal of relevance.
  - Query Precision:
    - Fraction of the top 5 recommendations that share a product with test query
  - Product Recall:
    - Fraction of products of test query that belong to at least one of the top 5 recommendations.
  - F1 score:
    - HM of QP and AR

Performance on Metrics

The row for theoretical best uses absolute values of the metrics.

All other rows are percentages of the theoretical best value.

Model	Query Precision	Product Recall	F1-score
Theoretical best	0.180	0.324	0.209
Attention	58.89%	67.90%	61.72%
Attention-RW	**59.44%**	**68.21%**	**62.20%**
Attention-Word	51.11%	59.26%	53.59%
Attention-Word-RW	52.22%	61.42%	55.02%
Attention-match-GS	1.11%	2.16%	1.44%
Trigram Hash	37.22%	51.58%	41.63%

Qualitative examples

- Queries that work well:
  - *red and balck hedadset with mic for gng* (gaming headset)
  - *fugo style* (FUGOO bluetooth speakers)
  - *tx-rz810* (audio receiver)
  - *dtse9h/16gb* (thumb drive)
- Queries that fail
  - *epson to601* (printer ink cartridges, but see printers)
  - *r7500* (WiFi router, but all models completely fail)
  - *unifi lr* (wireless access point, TrigramHash succeeds by chance)

The General problem

- **Training:** Text labelled nodes with edges to relate text entities.
- **Inference:** Only text available, edges with seen nodes are unknown.
- Text embeddings should mimic (unknown) underlying graph
  - Co-occurence à la word2vec is now represented as a graph
- Other concrete problems:

Queries and Product titles	Query to product or vice versa mapping
Product titles	Related products recommendation
Tweets	Find original author
StackOverflow titles	Related question recommendation

Theoretical justification overview

- Words, concepts and texts are isotropic vectors
- Text is a diffusive walk on words around a concept.
- However, words move in context of a concept.
- Attention can handle this modification.