Motivation

**Why ASIN?**

- **ProductQA Traffic Investigation**
  - Current retrieval system
    - Some answers found from top-10 ASINs
  - Rewriting questions to keyword queries
    - More can be answered by ASINs
Motivation

**ASIN is a good resource to answer product question**

- What is the name of the boat curved cutting knife from Alaska?
  - Rewrite to: **Alaska Knife**
    - **ALASKAN ULU KNIFE**: Title of the ASIN
      (www.amazon.com/dp/B006Z877C)

- What's the score available habanero pepper
  - Directly search using the proposed system
    - **100k-350k on the Scoville scale**: In the bullets of the ASIN (www.amazon.com/dp/B07SNGQHZ7)
Motivation

- **High precision**
  - Provide only frequently purchased/clicked ASINs

- **Naturally decide whether to answer based on ASIN**
  - When no result is similar enough to the PQ
    - ASIN is not suitable to the provided question
    - ASIN is not suitable to the provided question

- **Support Multimodality**
  - Retrieving ASINs beyond the use of text fields
Motivation

Token Matching IR v.s. Semantic Search System

- Token-based is unideal in retrieving ASINs
  - A gap between ASINs and Alexa questions
    - Terminology:
      - Specific product (ASINs)
      - General questions (PQs)
    - Format:
      - Include lots of info (ASINs)
      - Natural Language Questions (PQs)
Background —
Dense Vectors for Semantic Search

Male-Female
Verb tense
Country-Capital
Background — BERT
Bidirectional Encoder Representations from Transformer

• BERT considers contextual meaning

Use average representation from pre-trained BERT
Background —
Locality-Sensitive Hashing (LSH)

- Similar input items in the same "buckets"
- Search visits some buckets among all buckets
  - Reduce computation
Background — Locality-Sensitive Hashing (LSH)
Background —
Facebook AI Similarity Search (Faiss)

- A package for efficient similarity search
- One of the implementations is LSH
- “Bucket” is called “Voronoi cell” in Faiss
→ We use Faiss to retrieve ASINs
Overview

- Two components in the ASIN based PQA system
  - ASIN Retrieval System
  - ASIN Question Answering System
Method

**Offline System**

- Search Logs
  - Encoding (e.g., use BERT)
- Embedding Vectors
- Faiss Voronoi Cells
  - Indexing by LSH

**Online System**

- Alexa Product Questions
  - Encoding (e.g., use BERT)
- Embedding Vectors
- Faiss LSH Search
- Top-K similar ASINs
## Method

### Two Implementations on Buckets

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Index Loading</th>
<th>Inference</th>
<th>RAM Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>On Disk</td>
<td>Fast (mins)</td>
<td>Slow</td>
<td>Small (300 MB)</td>
</tr>
<tr>
<td>On RAM</td>
<td>Slow (hrs)</td>
<td>Fast</td>
<td>Huge (1.1TB)</td>
</tr>
</tbody>
</table>
Example – Offline System

- **Encoding**
  - Search Logs
    - Size: m
  - Embedding Vectors w/ ASINs
    - Size: (m, 768)

- **LSH Indexing**

```
Cell_1  
vector_100  
vector_20  
...  

Cell_2  
vector_593  
vector_3593  
...  

Cell_3  
vector_8  
vector_5289  
...  

...  

Cell_n  
vector_59  
vector_1523  
...  
```
Example — Online System

● **Product Question:**
  ○ What is the most up-to-date apple watch

● **Embedding Vector**
  ○ \[0.286, -0.426, ..., -0.368, -0.978]\]
  ○ Dimension: 768

● **Top-k similar ASIN(s)**
  ○ B07XR5T85R
  ○ Apple Watch Series 5 (GPS, 44mm) - with Silver Aluminum Case with White Sport Band
Evaluation – Hardware

● Offline Stage that indexes the (keyword, ASINs):
  ○ Four p2.16xlarge
    ■ GPUs: 16
    ■ vCPUs: 64
    ■ RAM (GiB): 732

● Online Stage that retrieves ASINs given a PQ:
  ○ One x1e.32xlarge
    ■ vCPUs: 128
    ■ RAM (GiB): 3904
Evaluation – Setup

- **Pre-trained Models Selection**
  - **Triplets:** \(<\text{PQ, True Title, Random Title}>\)
  - **Precision:** \[
  \frac{\sum \delta[L(E_{\text{PQ}}, E_{\text{true}}) < L(E_{\text{PQ}}, E_{\text{random}})]}{\sum \delta[\text{PQ}]} \]

<table>
<thead>
<tr>
<th>Model</th>
<th>Version</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>RoBERTa</td>
<td>roberta-base</td>
<td>80%</td>
</tr>
<tr>
<td>ELMo</td>
<td>small</td>
<td>82%</td>
</tr>
<tr>
<td><strong>BERT</strong></td>
<td>bert-base-uncased</td>
<td>90%</td>
</tr>
<tr>
<td>BERT</td>
<td>bert-base-multilingual-uncased</td>
<td>85%</td>
</tr>
</tbody>
</table>
Evaluation – Setup

● Efficiency:
  ○ Avg. time to retrieve top-5 ASINs for 923K questions
    (AmazonQA)

● Effectiveness:
  ○ Manual annotations (explained in the following slides)
  ○ 492 non-critical PQs
  ○ Only consider text fields in an ASIN
  ○ Relevance/answerability of top-1 retrieved ASIN per PQ
Evaluation — Annotation, Relevance

Query: Are sun dried tomatoes black?

Retrieved ASIN: B074MG6VGH

365 Everyday Value, Sundried Tomatoes, 4 oz
by Brand: 365 by Whole Foods Market
★ ★ ★ ★ ☆ 69 ratings
Currently unavailable.
We don't know when or if this item will be back in stock.

• Brought to you by Whole Foods Market. The packaging for this product has a fresh new look. During this transition, you may get the original packaging or the new packaging in your order, but the product and quality is staying exactly the same. Enjoy!
• Made without Sulfites.
• Fat Free, No Sugar Added.
Evaluation — Annotation, Answerable

Query: Are Triscuits crackers?

Retrieved

ASIN: B074XMX17J

Triscuit Dill, Sea Salt & Olive Oil Whole Grain Wheat Crackers, 8.5 oz
by Visit the Triscuit Store
★★★★★ 324 ratings
Price: $2.99 ($0.35 / Ounce) Pantry orders are delivered for $5.99 and do not qualify for FREE Delivery. Learn more

Pantry
Pantry orders of $35 and up ship for free- a $5.99 shipping fee applies to orders under $35. Due to high demand, delivery may take longer than usual. Learn more

In Stock.
Ships from and sold by Amazon Pantry.

Want it Thursday, Sep 10? Order within the next 19 hrs and 2 mins Details

- One 8.5 oz box of Triscuit Dill, Sea Salt & Olive Oil Whole Grain Wheat Crackers
- Dill, sea salt and olive oil flavors complement the whole grain wheat base
- **Square woven crackers** offer a classic, refined snacking option
- Baked, wholesome sea salt crackers with a hearty woven texture
- Savor the taste by itself or top with cheese or spreads
## Results — Index Construction

- **Time and Space Consumption for Index Construction**
  - Build the index per quarter before merging

<table>
<thead>
<tr>
<th></th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>On Disk</th>
<th>On RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Q</td>
<td>78M</td>
<td>80M</td>
<td>104M</td>
<td>115M</td>
<td></td>
<td>378M</td>
</tr>
<tr>
<td>Time (hr)</td>
<td>21.5</td>
<td>23.9</td>
<td>31.2</td>
<td>34.36</td>
<td>34.36</td>
<td>110.96</td>
</tr>
<tr>
<td>Size(GB)</td>
<td>224.5</td>
<td>231.5</td>
<td>299.2</td>
<td>330.1</td>
<td></td>
<td>1085.3</td>
</tr>
</tbody>
</table>
Results – Efficiency under Two Configurations

● Percent of Overall Data Touched / Search
  ○ 16 Buckets / 65536 Buckets → 0.02%

● Runtime (underestimated**)

<table>
<thead>
<tr>
<th>Type</th>
<th>Include Index Loading</th>
<th>Top-K</th>
<th>ms per question</th>
</tr>
</thead>
<tbody>
<tr>
<td>On Disk</td>
<td>No</td>
<td></td>
<td>68</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>5</td>
<td>69</td>
</tr>
<tr>
<td>On RAM</td>
<td>No</td>
<td></td>
<td>57</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td></td>
<td>65</td>
</tr>
</tbody>
</table>
Results – Efficiency under Two Configurations

● Why is it underestimated
  ○ Over 923K ASINs accumulated
  ○ Retrieved ASINs are stored on a list
  ○ Time consuming to maintain the list
**Results — Effectiveness**

- Experimented on different cut-off thresholds for distance

- The number of PQs have relevant retrieved ASINs @top-1

- The number of PQs have answerable retrieved ASINs @top-1

<table>
<thead>
<tr>
<th>Query</th>
<th>Retrieved ASINs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query 1</td>
<td>B007ZEMGS8 <em>(0.12)</em>, B078GZDJMH <em>(0.16)</em>, B00UW6EVAW <em>(0.27)</em></td>
</tr>
<tr>
<td>Query 2</td>
<td>B07RYB9G7Z <em>(0.25)</em></td>
</tr>
<tr>
<td>Query 3</td>
<td>B00LB4REF8 <em>(0.03)</em>, B0000EXU0S <em>(0.11)</em></td>
</tr>
</tbody>
</table>
Results — Effectiveness

#Q with ASINs vs. Distance Threshold

Distance Threshold

- 0.2: 5.30%
- 0.4: 15.40%
- 0.6: 37.40%
- 0.8: 66.90%
- 1.0: 88.60%
- 1.0: 95.70%
- 1.0: 100.0%

#Q with ASINs
Results – Effectiveness

#Q with Relevant/Answerable ASINs vs. Distance Threshold

- **#Q with relevant ASINs**
- **#Q with answerable ASINs**

![Graph showing the relationship between Distance Threshold and the number of queries with relevant and answerable ASINs.](image-url)
## Results — Effectiveness

<table>
<thead>
<tr>
<th>Distance Threshold</th>
<th>#Q with ASINs</th>
<th>#Q with relevant ASINs</th>
<th>#Q with ASINs including an answer</th>
<th>#Q including answer/#Q with ASIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&lt; 0.2</td>
<td>26 (5.3%)</td>
<td>21 (4.3%)</td>
<td>10 (2.0%)</td>
</tr>
<tr>
<td>2</td>
<td>&lt; 0.25</td>
<td>76 (15.4%)</td>
<td>43 (8.7%)</td>
<td>11 (2.2%)</td>
</tr>
<tr>
<td>3</td>
<td>&lt; 0.3</td>
<td>184 (37.4%)</td>
<td>95 (19.3%)</td>
<td>16 (3.3%)</td>
</tr>
<tr>
<td>4</td>
<td>&lt; 0.35</td>
<td>329 (66.9%)</td>
<td>138 (28%)</td>
<td>26 (5.3%)</td>
</tr>
<tr>
<td>5</td>
<td>&lt; 0.4</td>
<td>436 (88.6%)</td>
<td>159 (32.3%)</td>
<td>28 (5.7%)</td>
</tr>
<tr>
<td>6</td>
<td>&lt; 0.5</td>
<td>471 (95.7%)</td>
<td>161 (32.7%)</td>
<td>30 (6.1%)</td>
</tr>
<tr>
<td>7</td>
<td>N/A</td>
<td>492 (100%)</td>
<td>162 (32.9%)</td>
<td>30 (6.1%)</td>
</tr>
</tbody>
</table>
Result – Remaining Issues

- **QA system works only if ASINs include answers**
  - Some questions are not suitable to use ASIN as answer source
    - E.g. “Can my dog eat Cheetos”

- **Issue with our index on [Amazon.com](http://amazon.com) 2019**
  - Some ASINs may no longer exist
Future Work

- **Apply big data system** (e.g. Spark)
  - We are using 1-year data, there might be 10-year data
- **Collect product info related product questions**
  - Required to determine threshold
- **Some optimisations to further speed up**
  - Current runtime is ok, but preferably to be faster
  - Re-estimate the runtime
- **Encode PQs & keywords with supervised learning**
  - Current training dataset:

  ![BookCorpus](http://example.com/)
  ![English Wikipedia](http://example.com/)