Machine Learning @ Amazon

Rajeev Rastogi

Director, Machine Learning

amazon

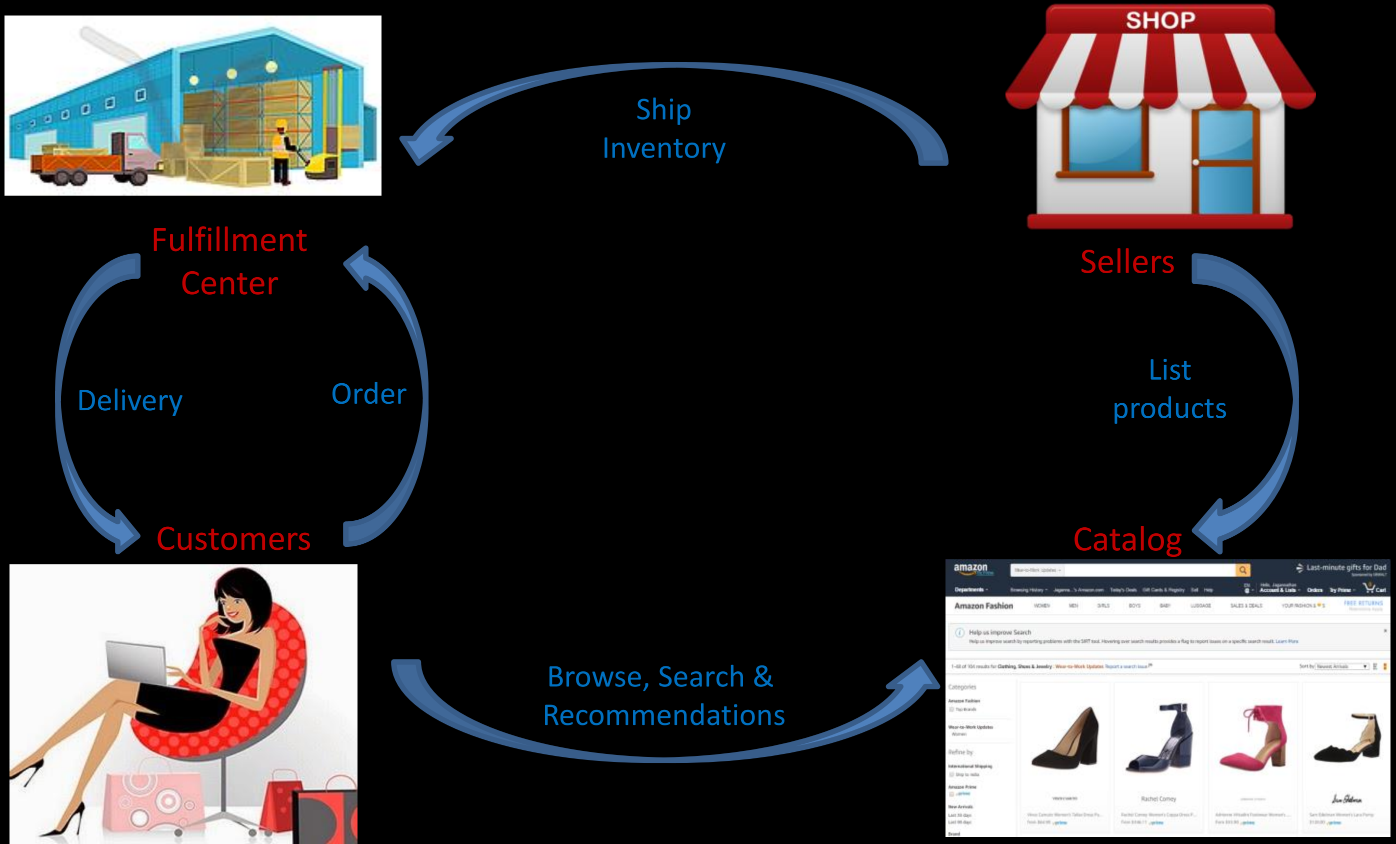
Key Takeaways

- ML is enabling smart ecommerce
  - Product recommendations, demand forecasting, search, classification, matching, …
- Learning semantically rich representations critical for many current tasks and future AI applications
  - Online advertising, product search, **question answering**, **product recommendations**, fake reviews detection
  - Conversational systems, video metadata generation, content summarization
- Several techniques for learning representations
  - **Deep Learning**, **Probabilistic Graphical Models**, Tensor Decomposition
  - Leverage diverse signals/data

Outline

- Applications of ML @ Amazon
- Question answering
- Product size recommendations

Amazon Marketplace



amazon

Using ML to Address Top Business Priorities

- Increasing product selection
- Lowering prices
- Reducing delivery times
- Eliminating friction
- Maintaining customer trust

Numerous ML Applications

Retail
- Demand Forecasting
- Vendor Lead Time Prediction
- Pricing
- Packaging
- Substitute Prediction

Customers
- Product Recommendation
- Product Search
- Product Ads
- Customer Targeting

Sellers
- Seller Nudges
- Fraud Detection
- Predictive Help
- Seller Lead Generation

Catalog
- Product Classification
- Product matching
- Reviews Ranking and Analysis
- Fraudulent Reviews detection

Text
- Named-Entity Extraction/Xray
- Summarization
- In-Book Search
- Plagiarism Detection

Images
- Visual Search
- Product Image Quality
- Brand Tracking

AWS
- Predicting Resource Requirements
- DDoS detection
- Reputation computation of Mturk workers

Speech
- Automatic speech recognition
- Natural Language Understanding
- Dialog Management

amazon

Product Demand Forecasting

Problem

- *Given past sales of a product in every region of the world, predict regional demand up to one year into the future*

Product Demand Forecasting

Problem

- *Given past sales of a product in every region of the world, predict regional demand up to one year into the future*

Challenges

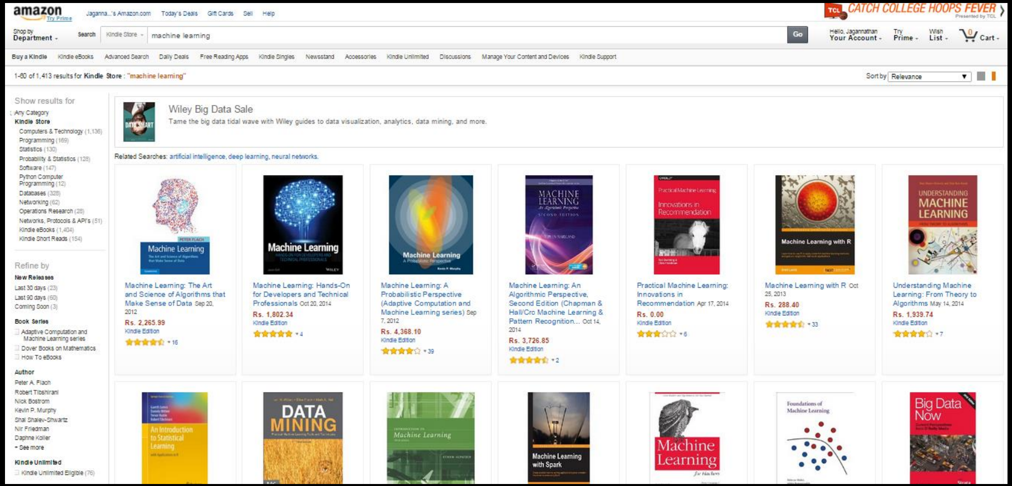
- **Scale**: Hundreds of millions of products in Amazon catalog!
- **New products**: No past demand!
- **Sparsity**: Huge skew – many products sell very few items
- **Seasonal**: Demand for some products exhibits seasonal patterns
- **Demand spikes**: Huge variation due to external events
- **Distributions**: Future is uncertain → predictions must be distributions

amazon

Product Search

Problem

- *Given a partial user query, find the relevant products to display to user in search results*



Product Search

Problem

- *Given a partial user query, find the relevant products to display to user in search results*

Challenges

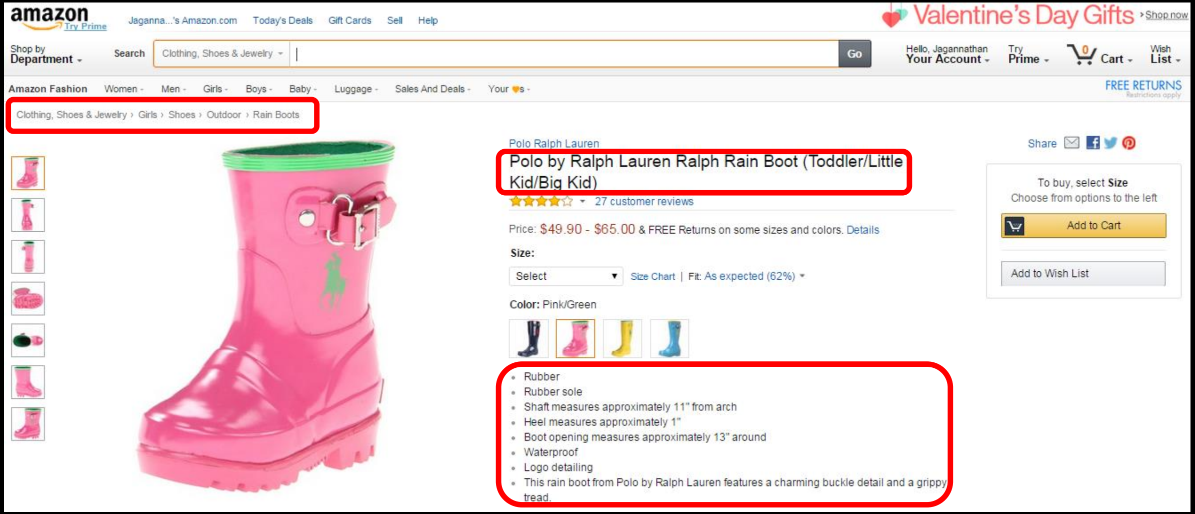
- **Scale**: Hundreds of millions of products in Amazon catalog
- **Real-Time Prediction**: Search requires low-latency (<40ms)
- **Query analysis:** Understand semantics, identify phrases, classify into product category (e.g. red apple iphone vs red apples)
- **Intent detection**: Is customer researching product or looking to buy product
- **Knowledge Graph**: Structured entity data + related products in response to search queries

amazon

Product Classification

Problem

- *Given a product description from a seller, map it to the appropriate leaf node in product taxonomy*



Product Classification

Problem

- *Given a product description from a seller, map it to the appropriate leaf node in product taxonomy*

Challenges

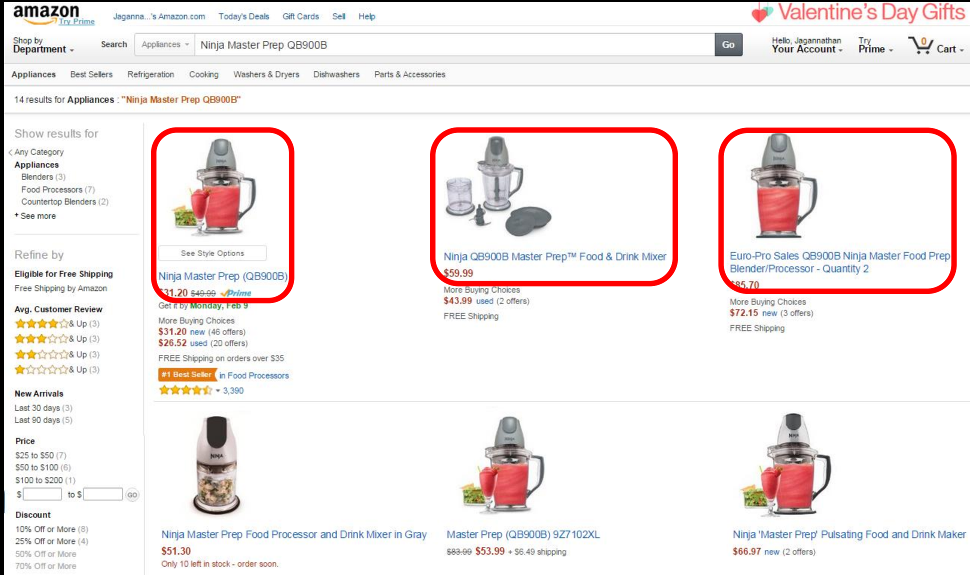
- **Scale:** Hundreds of millions of products and thousands of classes in product taxonomy
- **Products vs Accessories:** Hard to distinguish between products and its accessories (e.g. laptop vs laptop battery)
- **Incorrect/Missing data:** Some attribute values may be wrong, missing or inadequate (e.g. short titles)
- **Training data quality:** Examples may be wrongly labeled, some classes may have very few examples

amazon

Product Matching

Problem

- *Given product information (title, description, price, etc.), find duplicate product listings in Amazon catalog*



Product Matching

Problem

- *Given product information (title, description, price, etc.), find duplicate product listings in Amazon catalog*

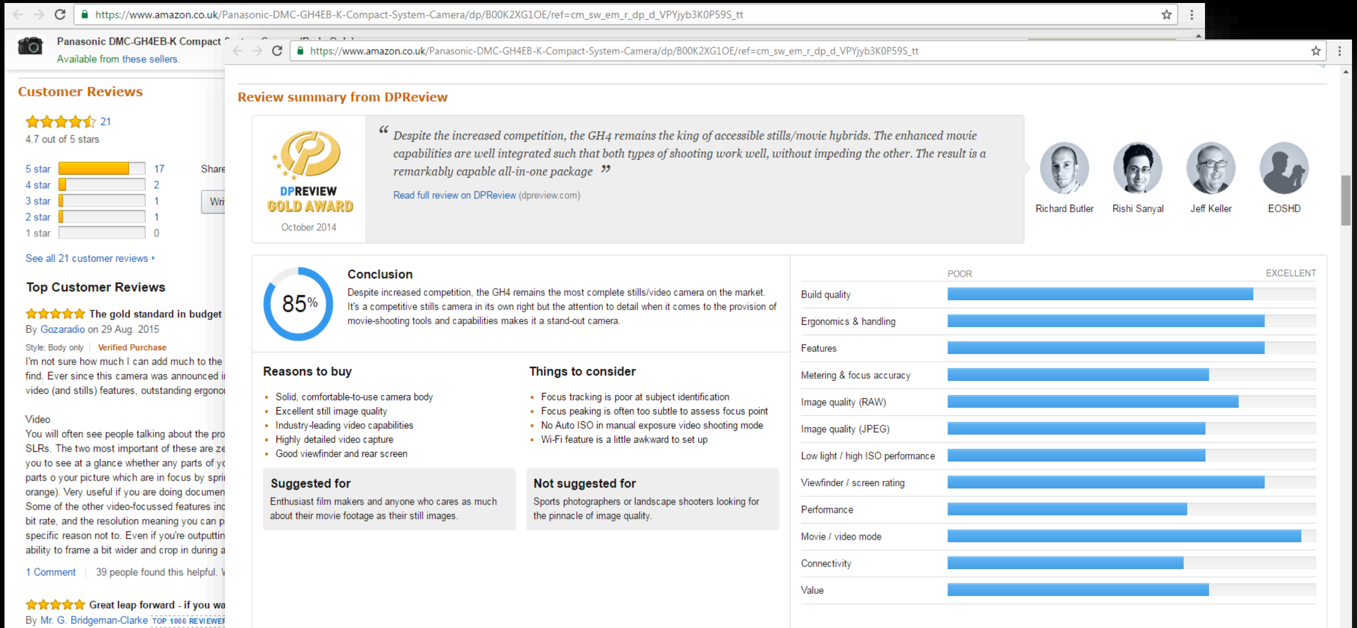
Challenges

- **High-Precision requirement:** Incorrect matching leads to poor user experience
- **Variations:** Some variations (e.g., color) are insignificant while others (e.g., movie sequels) are not
- **Diverse formats**: Attribute values for the same product may be represented differently
- **Incorrect/Missing Data:** Some attribute values may be missing or wrong in product descriptions

Information Extraction from Reviews

Problem

- *Generate review summaries, extract product attribute ratings from reviews*



Information Extraction from Reviews

Problem

- *Generate review summaries, extract product attribute ratings from reviews*

Challenges

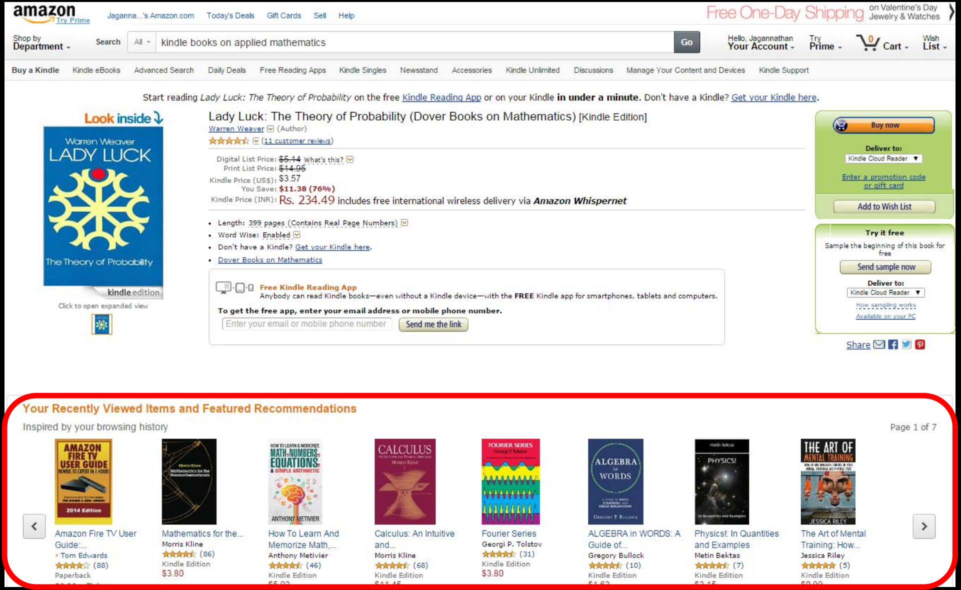
- **Diverse attributes:** Product attributes may vary across different categories
- **Synonyms:** Different terms may refer to same attribute (sound, audio)
- **Informal style:** User reviews have an informal linguistic style
- **Stylistic variations:** Linguistic style varies between users
- **Sentiment analysis**: Gauging sentiment may require deep parsing of sentences

amazon

Product Recommendations

Problem

- *Discover products by recommending the right product, to the right customer, in the right place, at the right time*



Drones

Problem

- *Safely deliver packages to customers homes within 30 minutes using drones*

amazon

Robotics

Problem

- *Automate picking, stowing and transport of products in Amazon Fulfillment Centers*

amazon

Visual Search

Problem

- *Retrieve images from the Amazon catalog that are visually similar to a given product*

Voice Recognition

Problem

- *Provide a voice interface to shop for products, perform tasks, answer questions, and carry out conversations*

The Amazon Echo family

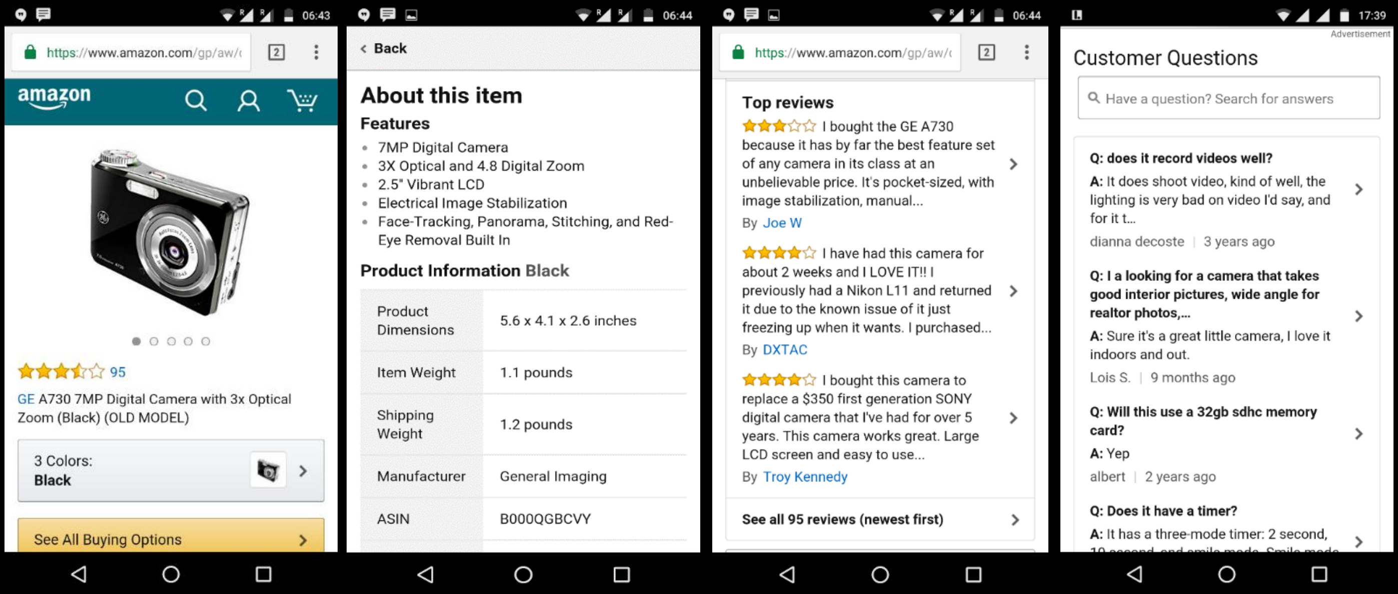
amazon

Outline

- Applications of ML @ Amazon
- Question answering
- Product size recommendations

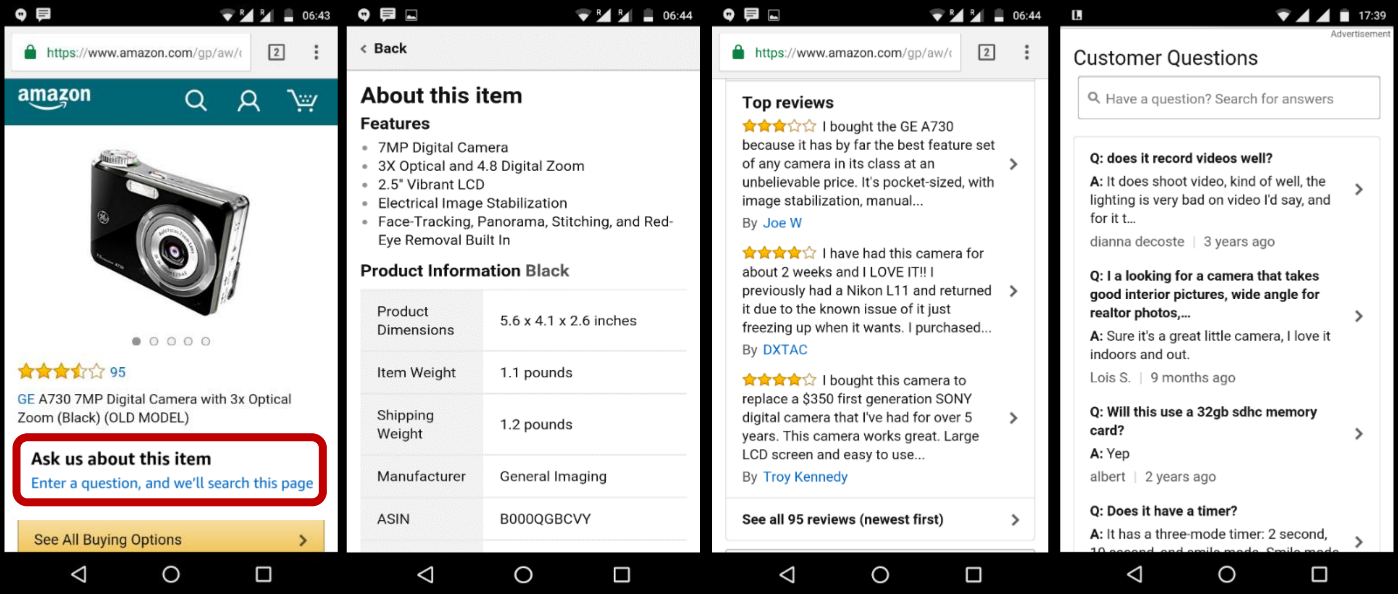
Amazon Product Pages

- Amazon product pages contain a wealth of information

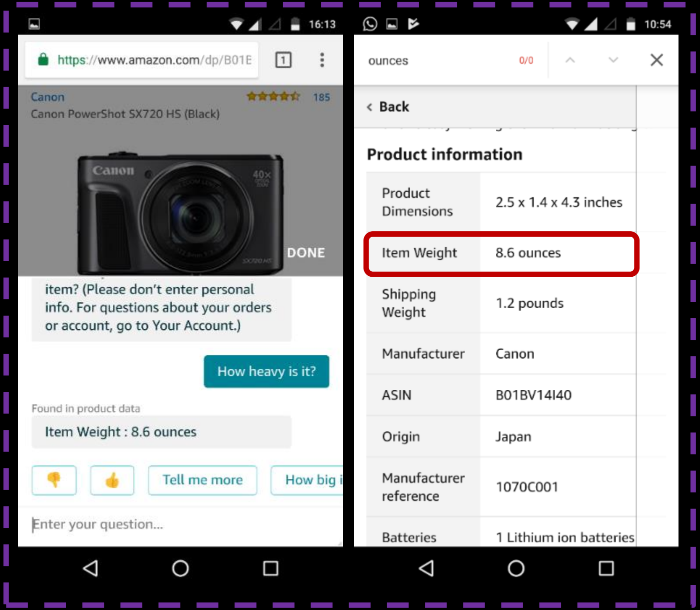


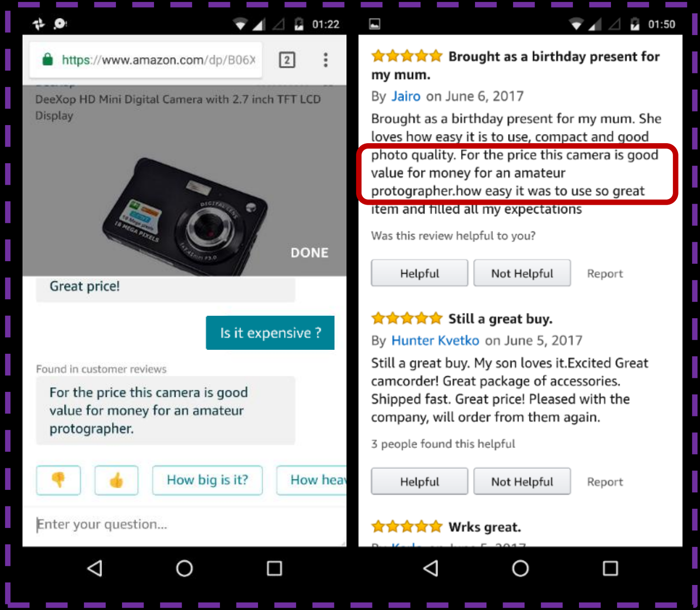
Question & Answering Bot

- Question answering interface to make it easy for users to find information on product page

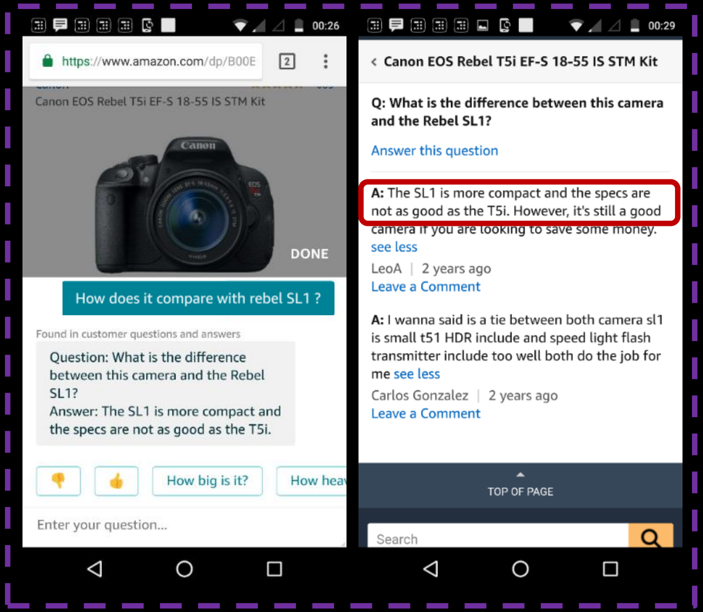


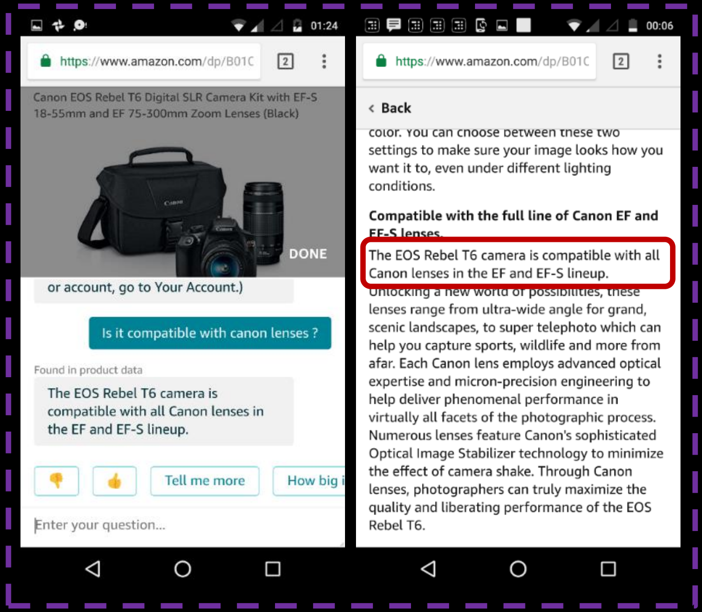
Product Feature Questions





Product Comparison/Compatibility Questions



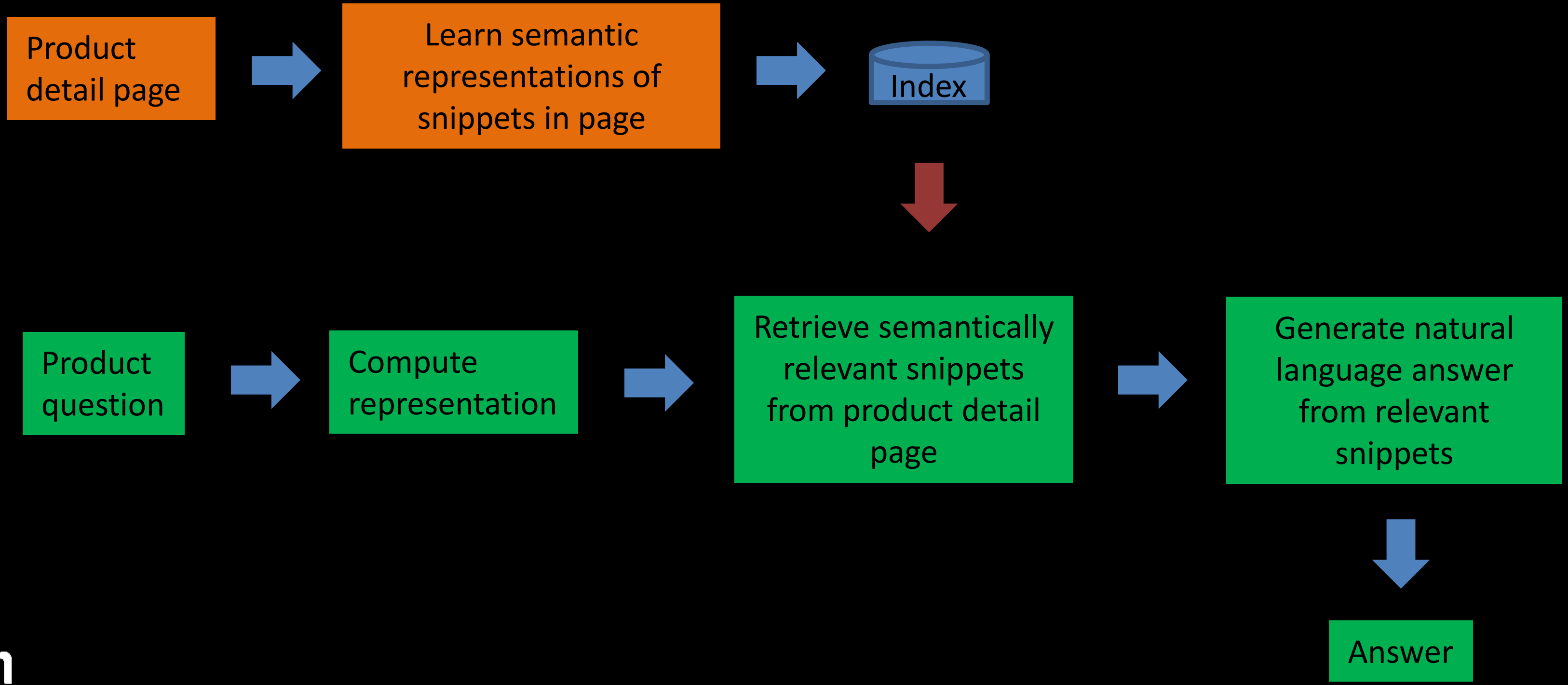


amazon

Key Challenges

- Question understanding
  - “What is ISO?” vs “What is the ISO [of this camera]?”
- Semantic matching
  - “cost”, “price”, “bang for buck”, “expensive”, “cheap”
- Natural language answer generation
  - e.g. “This is great value for money” for question “Is this expensive?”
- High precision (>90%) requirement
- Data availability
  - “Will this suitcase fit in the overhead of an airplane?”
- Data quality
  - “Dimensions: 1x1x1 inches” for the question “How big is it” on a “Tripod page”

Question Answering System Architecture

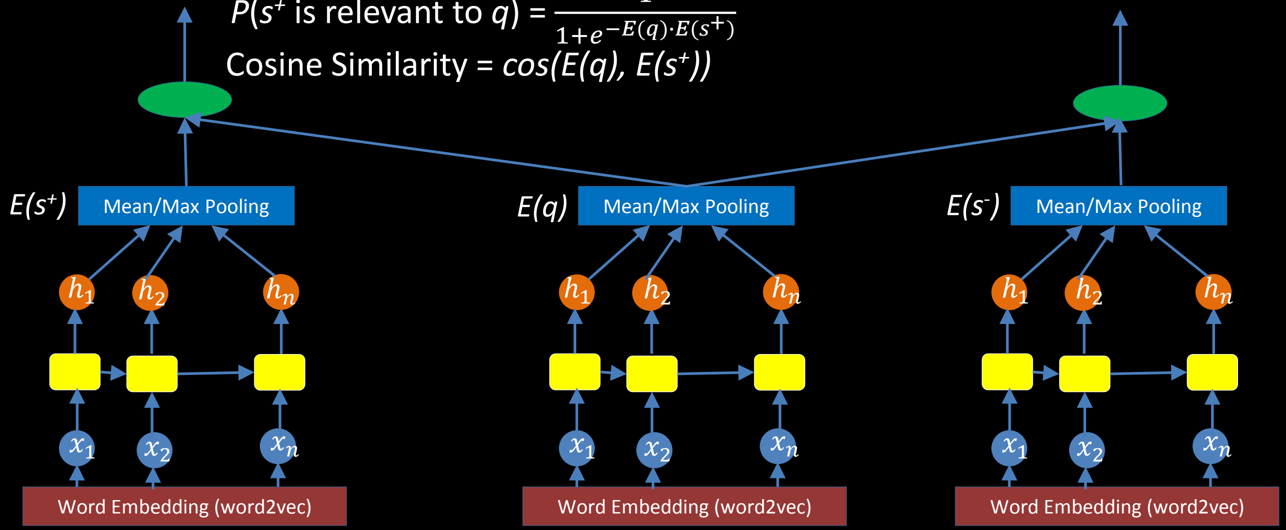


Learning Semantically Rich Representations

- Training examples: <question (*q*), relevant snippet (s^+), irrelevant snippet (s^-)> triples
- **Triplet network**

$P(s^+ \text{ is relevant to } q) = \frac{1}{1+e^{-E(q)\cdot E(s^+)}}$

Cosine Similarity = $cos(E(q), E(s^+))$



s^+: Product is very good for the given price

q: Is it expensive?

s^-: Auto focus is great

amazon

Results for Different Loss Functions

- Learn question and snippet representations to minimize the following loss functions:
  - Log loss [EMNLP 2015]
    $$-\log\frac{1}{1+e^{-E(q)\cdot E(s^+)}}-\log\frac{1}{1+e^{E(q)\cdot E(s^-)}}$$
  - Siamese loss [Wang et al. 2014]
    $$\max\{0, \mathrm{M}-(\cos\big(E(q),E(s^+)\big)-\cos(E(q),E(s^-)))\}$$
  - Twin loss
    $$\max\{0, M_1-\cos\big(E(q),E(s^+)\big)\}+\max\{0,\cos\big(E(q),E(s^-)\big)-M_2\}$$
- Metric: Precision at rank 1 (P@r1)
- Results:

Loss function	Baseline	Log loss	Siamese loss	Twin loss
P@r1	56.8%	84.6%	96.1%	97.04%

[Wang et al 2014] Learning fine-grained image similarity with deep ranking, CVPR 2014.

Qualitative Results

Question	Matching Snippet
Is this camera good for pictures at a basketball game?	Works great for **sports** photography
What is the price ?	This item **costs** $100.00. To see tax and shipping, add to cart
How big is it?	Item **dimensions**: 3 x 3.28 x 4.37 inches
How good is stabilization?	EVERY image came out **blurry** (and I held the camera still in a well-lit room).
Will it fit on Olympus air?	Fits very well the Olympus Air OA-01
How much weight can it hold?	Item weight: 2.2 pounds
What is the color of the paper on which the photo is printed?	the color of the camera and the pictures are great.

Training Dataset Generation

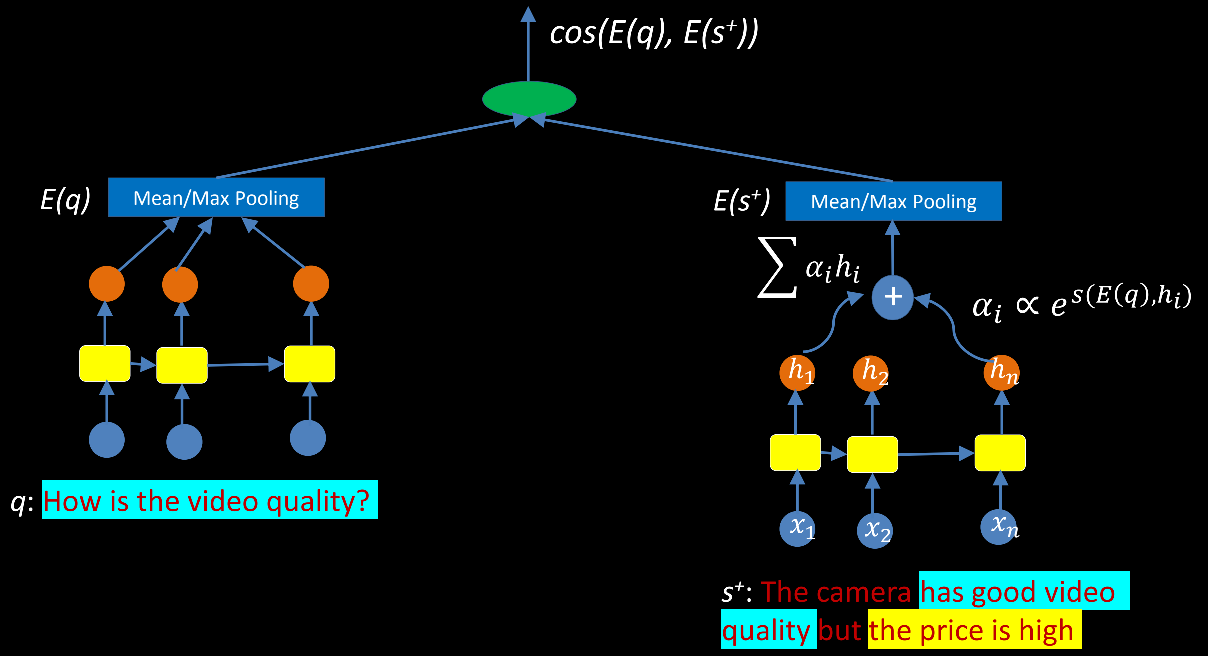
- Raw Data
  - Corpus of Amazon customer reviews, product descriptions, community QA
  - Total size is 147.6M records comprising 3.68M word tokens
  - Trained word2vec embeddings of 200 dimensions

- Training examples: triplets (q, s^+, s^-)
  - Attribute selection: mined attributes using term frequency (~130 for cameras)
    - price, image stabilization, zoom, resolution, etc.
  - Question templates (q) for attributes
    - price : “what’s the price”, “what is the cost”, etc.
  - Relevant snippets (s^+): defined set of matching words
    - price: “price”, “expensive”, “cheap”, “worth”, “money”, “pricey”, “investment”, “buck”, “cheaply”
  - Irrelevant snippets (s^-): used negative sampling
  - Over 10M triplets

amazon

Learning Representations with Attention

- Only consider relevant portions of snippets when learning representations [Bahdanau et al. 2015]



[Bahdanau et al. 2015] Neural machine translation by jointly learning to align and translate, ICLR 2015.

Highlighting Words with High Attention Weights

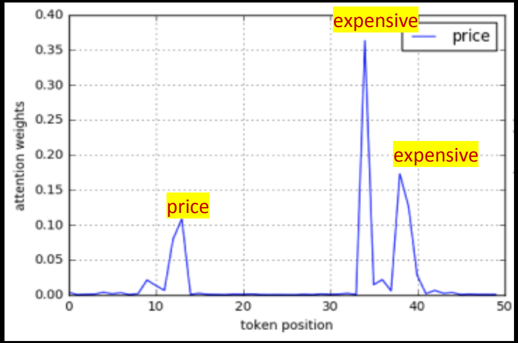
Review Statement:

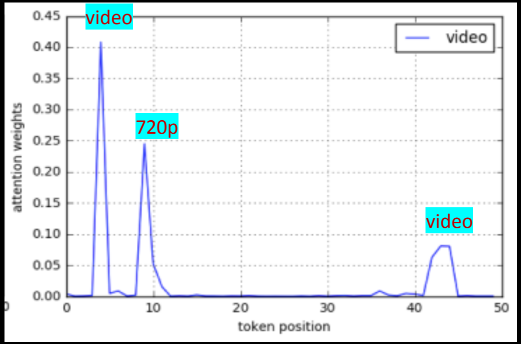
The package is good video quality is good for 720p but the price is excellent for what you get - especially if you do not want all the whistles and bells of the more expensive gopro 2-4x more expensive and the quality of video is superb and great.

Questions:

Question1: What is the price?

Question2: How good is the video?

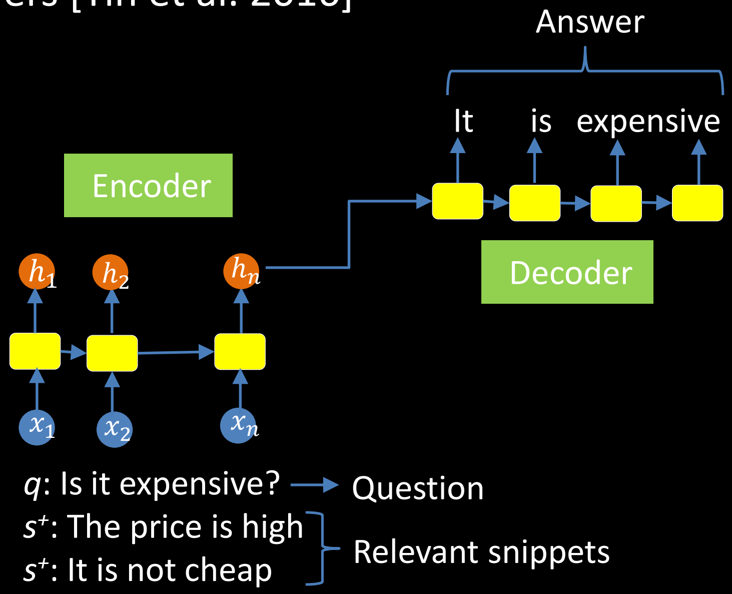




amazon

Generating Natural Language Answers

- Use sequence-to-sequence encoder-decoder model to generate final answers [Yin et al. 2016]



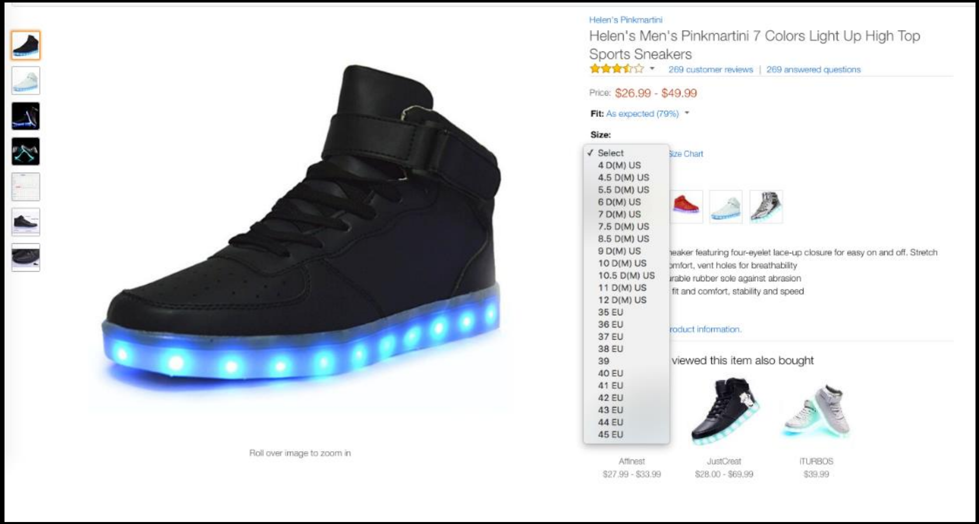
[Yin et al. 2016] Neural generative question answering. IJCAI 2016.

Outline

- Applications of ML @ Amazon
- Question answering
- Product size recommendations

Size Recommendation Problem

- Given a customer and product, recommend product size that would best fit the customer



Motivation

- No standardization of sizes across brands and locales for product categories such as shoes and apparel
- This leads to users making incorrect purchases, and then returning products
- Products belonging to shoe and apparel categories have high return rates due to fit issues
- Example:
  - Reebok size mapping convention: 6 = 15cm, 7 = 17cm, 8 = 21cm
  - Nike size mapping convention: 6 = 16cm, 7 = 18cm, 8 = 22cm

amazon

Key Challenges

- Scale: hundreds of millions of customers and products
- Data sparsity: bulk of users/products have very few purchases
- Cold start scenarios: new customers/products
  - User features: demographics (age, gender), location
  - Product features: catalog size, title, brand, product type
- Multiple personas: each customer *i* may involve multiple personas
  - E.g., family members sharing an account
  - Personas may have widely varying sizes

Our Approach

- Learn true (latent) size for each customer, product
  - True size for customer corresponds to the physical size of the customer (for shoes, it would be the feet size)
  - True size for product corresponds to it's physical size
- Leverage past customer transactions $T = \{(i, j, y_{ij})\}$
  - y_{ij} takes ordinal values {small, fit, large}

	Adidas (9)	Nike (8)	Reebok (8)	Nike (9)
Customer 1	large		fit	?
Customer 2		small		fit
Customer 3	fit		small	?
Customer 4		fit		large

Catalog size

Predict fit outcome

amazon

Our Approach (Contd)

- Notation
  - Latent size for customer *i*: s_i
  - Latent size for product *j*: t_j
  - Catalog size for product *j*: c_j

- Model likelihood of fit as a function of the difference between customer and product latent sizes

$$P(y_{ij} = fit) \propto f(s_i - t_j)$$

- Recommend product *j* with highest fit likelihood $P(y_{ij} = fit)$ to customer *i*

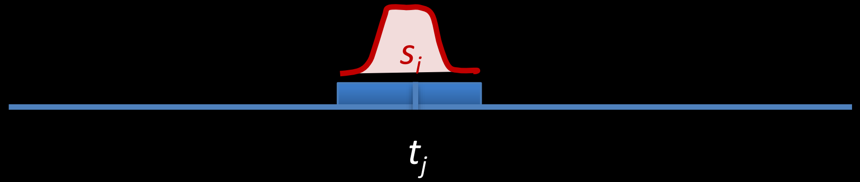
Bayesian Modeling Benefits

- Handles data sparsity by placing priors on latent size variables
- Models uncertainty in inferred latent sizes
  - Estimates posterior distribution of latent size variables
  - Fit probability is obtained by averaging over posterior size distribution
- Model can capture all the available data
  - Observations: transaction outcomes, customer and product features
  - Hidden variables: latent sizes, customer personas
- Efficient techniques for approximating posterior distributions of latent size variables

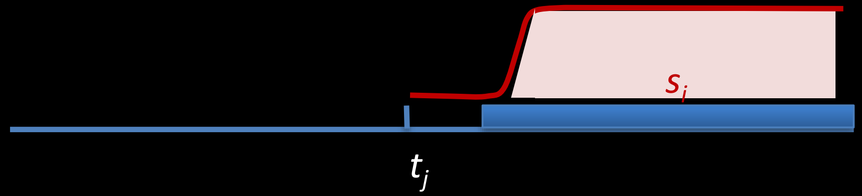
amazon

Intuition

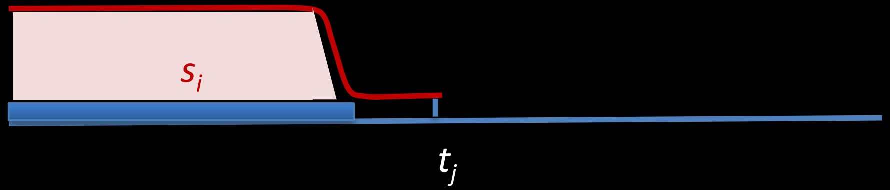
- Transaction (*i*, *j*, *fit*) → very likely that s_i and t_j are close



- Transaction (*i*, *j*, *small*) → very likely that s_i is much larger than t_j



- Transaction (*i*, *j*, *large*) → very likely that s_i is much smaller than t_j



Data Likelihood

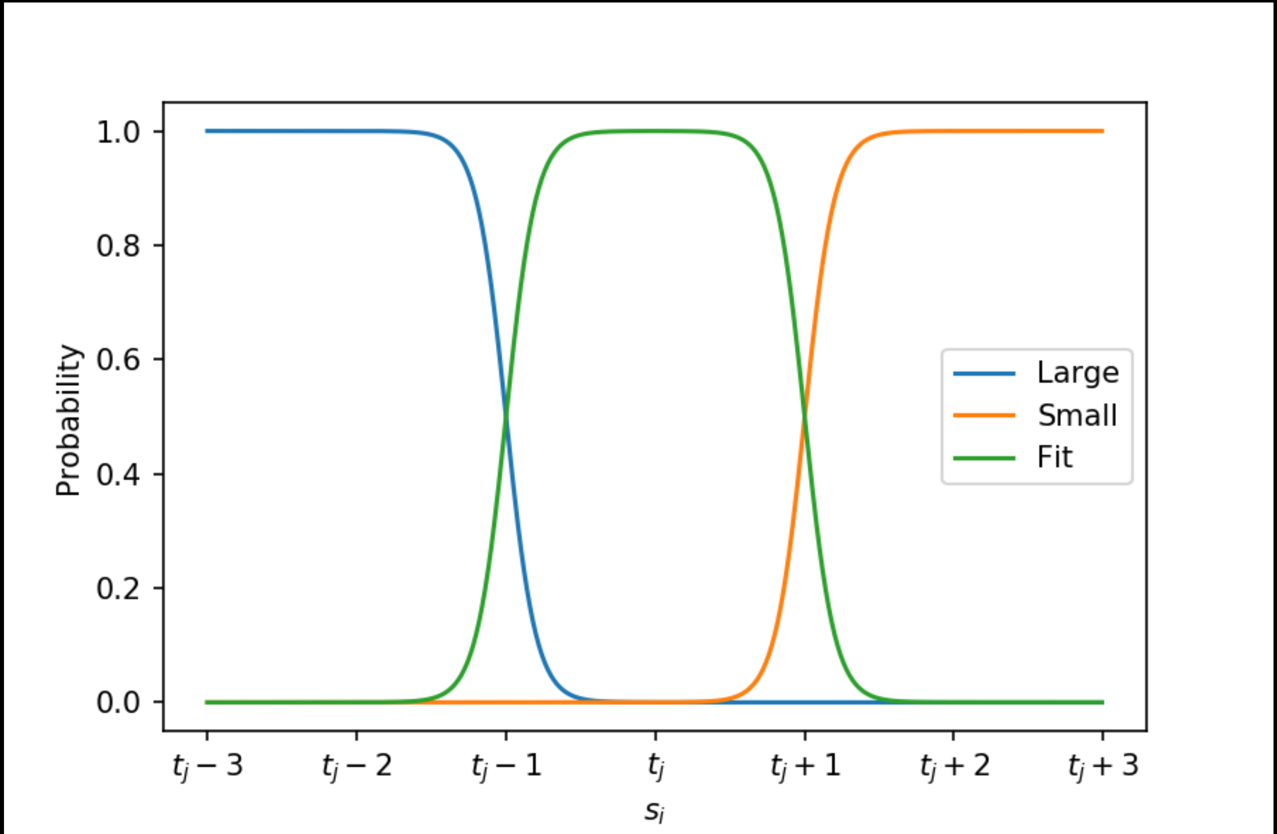
$$P\left(y_{ij} = small|s_i, t_j\right) = \frac{1}{1+e^{-\alpha\left(s_i-t_j\right)+b_1}}$$

$$P\left(y_{ij} = fit|s_i, t_j\right) = \frac{1}{1+e^{\alpha\left(s_i-t_j\right)-b_1}} \cdot \frac{1}{1+e^{-\alpha\left(s_i-t_j\right)+b_2}}$$

$$P\left(y_{ij} = large|s_i, t_j\right) = \frac{1}{1+e^{\alpha\left(s_i-t_j\right)-b_1}} \cdot \frac{1}{1+e^{\alpha\left(s_i-t_j\right)-b_2}}$$

[Albert & Chib 1993] Bayesian analysis of binary and polytomous response data, JASA 1993.

Data Likelihood



Generative Model

for each customer i,

 draw latent size $s_i \sim N(\mu_i, \sigma_s^2)$

for each product j,

 draw latent size $t_j \sim N(c_j, \sigma_t^2)$

for each transaction $(i, j, y_{ij}) \in T$,

 select $y_{ij} = small$ with probability $P(y_{ij} = small | .)$

 select $y_{ij} = fit$ with probability $P(y_{ij} = fit | .)$

 select $y_{ij} = large$ with probability $P(y_{ij} = large | .)$

amazon

Bayesian Inference

- Let β be the vector of latent sizes

$$\beta = \left(s_1, \dots, s_c, t_1, \dots, t_p, 1\right)^T$$

- Posterior distribution

$$P(\beta|T) \propto P(T|\beta) \cdot P(\beta)$$

$$\propto \prod_{(x,y)\in \boldsymbol{D}} \frac{e^{y\beta^T \cdot x}}{1 + e^{\beta^T \cdot x}} \cdot \prod_i N(s_i|\mu_i, \sigma_s) \cdot \prod_j N(t_j|c_j, \sigma_t)$$

0/1

$$(0, \dots, 0, \alpha, 0, \dots, 0, -\alpha, 0, \dots, 0, -b_1/b_2)^T$$

- Not available in closed form due to logistic likelihood terms and Normal priors

Polya-Gamma Augmentation [Polson et al. 2013]

- Introduce Polya-Gamma latent variable $w \sim PG(0,1)$ for every $(x, y) \in \boldsymbol{D}$
- Define the joint likelihood distribution

$$P(w, y|x, \beta) = \frac{1}{2} e^{\left(\left(y-\frac{1}{2}\right)\cdot(\beta^T \cdot x) - w \cdot \frac{\left(\beta^T \cdot x\right)^2}{2}\right)} \cdot P(w)$$

- In [Polson et al. 2013], it is shown that

$$\int_0^\infty P(w, y|\beta, x) dw = \frac{e^{y\beta^T \cdot x}}{1 + e^{\beta^T \cdot x}}$$

amazon

[Polson et al. 2013] Bayesian inference for logistic models using Polya-Gamma latent variables, JASA 2013.

Polya-Gamma Augmentation (Contd)

- Let W be the set of Poly-Gamma variables w for $(x, y) \in \boldsymbol{D}$

$$P(\beta|\boldsymbol{D}) \propto \int P(W, \boldsymbol{D}|\beta) \cdot P(\beta) dW$$

- Approximate the augmented joint distribution $P(W, \boldsymbol{D}|\beta) \cdot P(\beta)$

$$\prod_{(x,y)\in\boldsymbol{D}} \frac{1}{2} e^{\left(\left(y-\frac{1}{2}\right)\cdot(\beta^T\cdot x) - w\cdot\frac{(\beta^T\cdot x)^2}{2}\right)} \cdot P(w) \cdot \prod_i N(s_i|\mu_i, \sigma_s) \cdot \prod_j N(t_j|c_j, \sigma_t)$$

amazon

Gibbs Sampling Algorithm

- Conditional distribution of w_i

$$P(w_i|\beta, \boldsymbol{D}) \propto e^{-w_i \frac{(\beta^T \cdot x_i)^2}{2}} P(w_i)$$

$$= PG(w_i|1, \beta^T \cdot x_i)$$

- Conditional distribution for β_j

$$P\left(\beta_j|\beta_{-j}, W, \boldsymbol{D}\right) \propto \prod_{(x_i,y_i)\in \boldsymbol{D}} \prod_{x_{ij}\neq 0} \frac{1}{2} e^{\left(\left(y_i - \frac{1}{2}\right)\cdot(\beta^T \cdot x_i) - w_i \cdot \frac{(\beta^T \cdot x_i)^2}{2}\right)} N(\beta_j|\mu_{\beta_j}, \sigma_{\beta_j})$$

$$= N(\beta_j|m_j, V_j)$$

$$m_j = V_j\left(\frac{\mu_{\beta_j}}{\sigma_{\beta_j}^2} + \sum_{(x_i,y_i)\in \boldsymbol{D}} \sum_{x_{ij}\neq 0} \left(\left(y_i - \frac{1}{2}\right) \cdot x_{ij} + w_i \cdot x_{ij} \cdot \sum_{l\neq j} \beta_l \cdot x_{il}\right)\right)$$

$$\frac{1}{V_j} = \frac{1}{\sigma_{\beta_j}^2} + \sum_{(x_i,y_i)\in \boldsymbol{D}} \sum_{x_{ij}\neq 0} w_i \cdot x_{ij}^2$$

Predictive Distribution

- Let r samples be drawn from the posterior of β
- Let the l^{th} sample be $\beta^l = (s_1^l, \ldots, s_c^l, t_1^l, \ldots, t_p^l)$

$$P\left(y_{ij} = fit \middle| \boldsymbol{D}\right) = \int P\left(y_{ij} = fit \middle| \beta\right) \cdot P(\beta|\boldsymbol{D})d\beta$$

$$\approx \frac{1}{r}\sum_{l=1}^{r} \frac{1}{1+e^{\alpha\left(s_i^l - t_j^l\right) - b_1}} \cdot \frac{1}{1+e^{-\alpha\left(s_i^l - t_j^l\right) + b_2}}$$

amazon

Experimental Results

- Consider 6 real-life shoes datasets with between 10M and 33M transactions
- Baseline model
  - Product size $t_j = c_j$
  - Customer size s_i = Average size of products purchased by customer
  - Logistic regression model with feature (s_i-t_j) to predict outcome
- Bayesian Logit model
  - Predict outcome with highest probability
- Performance metric: weighted AUC
- Results (% improvements over baseline)

Dataset	A	B	C	D	E	F
Bayesian	17.71	18.28	19.7	25.78	20.22	19.42

amazon

Leveraging Customer and Product Features

- Means of latent size priors are obtained by performing regression over customer (f_i) and product (g_j) features [AC 2009]
- **Generative model:**

for each customer i,
    draw latent size $s_i \sim N(w_f \cdot f_i, \sigma_s^2)$
for each product j,
    draw latent size $t_j \sim N(w_g \cdot g_j, \sigma_t^2)$
for each transaction $(i, j, y_{ij}) \in T$,
    select $y_{ij} = small$ with probability $P(y_{ij} = small | .)$
    select $y_{ij} = fit$ with probability $P(y_{ij} = fit | .)$
    select $y_{ij} = large$ with probability $P(y_{ij} = large | .)$

- Perform least squares regression to learn parameters w_f and w_g from customer and product size samples

amazon

[Agarwal & Chen 2009] Regression based latent factor models, KDD 2009.

Incorporating Customer Personas

- Latent size for persona *k* of customer *i*: s_{ik}
- Latent variable containing persona involved in transaction (i, j, y_{ij}): z_{ij}
- **Generative model:**

for each customer *i*,
  draw persona distribution θ_i ~ Dir(α)
  for each persona *k* draw latent size $s_{ik} \sim N(w_f \cdot f_i, \sigma_s^2)$
for each product *j*,
  draw latent size $t_j \sim N(w_g \cdot g_j, \sigma_t^2)$
for each transaction $(i, j, y_{ij}) \in T$,
  draw persona z_{ij} ~ Mult(θ_i)
  select $y_{ij} = small, \ldots$ with probability $P(y_{ij} = small | z_{ij}), \ldots$

- Gibbs Sampling algorithm can be extended to draw z_{ij} samples

Summary

- Learning semantically rich representations critical for future AI applications, several ML techniques
  - Conversational systems, content summarization, video metadata generation
  - Deep Learning, Probabilistic Models, Tensor Factorization
- **Deep Learning** to learn embeddings
  - Allows semantic matching between questions and snippets
  - Loss functions like Siamese loss that aim to maximize difference in class scores perform better
- **Probabilistic Graphical Models** to learn latent sizes
  - Priors handle data sparsity, prevent overfitting
  - Posteriors model uncertainty in data
  - Leverage all the available signals