ANTHEM: Attentive Hyperbolic Entity Model for Product Search

Nurendra Choudhary¹, Nikhil Rao², Sumeet Katariya², Karthik Subbian², Chandan K. Reddy¹,²
¹Department of Computer Science, Virginia Tech, Arlington, VA, USA
²Amazon, Palo Alto, CA, USA
nurendra@vt.edu, {nikhilr, katsumeeka, ksubbianj}@amazon.com, reddy@cs.vt.edu

ABSTRACT
Product search is a fundamentally challenging problem due to the large-size of product catalogues and the complexity of extracting semantic information from products. In addition to this, the black-box nature of most search systems also hamper a smooth customer experience. Current approaches in this area utilize lexical and semantic product information to match user queries against products. However, these models lack (i) a hierarchical query representation, (ii) a mechanism to detect and capture inter-entity relationships within a query, and (iii) a query composition method specific to e-commerce domain. To address these challenges, in this paper, we propose an AtteNTive Hyperbolic Entity Model (ANTHEM), a novel attention-based product search framework that models query entities as two-vector hyperboloids, learns inter-entity intersections and utilizes attention to unionize individual entities and inter-entity intersections to predict product matches from the search space. ANTHEM utilizes the first and second vector of hyperboloids to determine the query’s semantic position and to tune its surrounding search volume, respectively. The attention networks capture the significance of intra-entity and inter-entity intersections to the final query space. Additionally, we provide a mechanism to comprehend ANTHEM and understand the significance of query entities towards the final resultant products. We evaluate the performance of our model on real data collected from popular e-commerce sites. Our experimental study on the offline data demonstrates compelling evidence of ANTHEM’s superior performance over state-of-the-art product search methods with an improvement of more than 10% on various metrics. We also demonstrate the quality of ANTHEM’s query encoder using a query matching task.

CCS CONCEPTS
• Information systems → Retrieval models and ranking; Query representation; • Applied computing → Online shopping.

KEYWORDS
E-commerce, product search, query representation, hyperbolic space

1 INTRODUCTION
In e-commerce, a majority of customers begin their journey on websites via a product search functionality. When a user searches for an item, they would potentially see a ranked (or tiled) list of products that best match the intent of the query. User queries are often vague, broad, short, and do not follow any specific natural language structure. Additionally, the catalogue for e-commerce websites is ever growing, and rapidly changing. The above reasons compelled with the need to show an array of related, yet complementary and substitute items makes it hard to match and show appropriate results to these queries.

Product search heavily influences both user behavior and experience. Not only do unsuitable results upset user experience, but the black-box nature of current search systems does not allow researchers to gain insight into the problems of querying process. Thus, the only source of feedback is the final set of search results. Being able to interpret these search results will allow researchers to gain insight into the system and subsequently improve both their query comprehension and query processing methods [10, 24, 33].

Figure 1: Product search framework. This paper focuses on improving the Product Matching module, optimized for recall in semantic matching and precision in ranking.

Current search frameworks, as shown in Figure 1, include two major modules for retrieving the product matches for a given input query [28]: (i) a matching phase that generates a set of items deemed appropriate to the query, and (ii) a ranking phase that ranks these items in a certain order of suitability. Traditional approaches for matching [21, 41] lexically match queries to an inverted index to retrieve all products that contain the query’s words. Such methods do not understand the query’s semantic intent of hypernyms (sneakers vs running shoes), synonyms (blue vs sapphire) and antonyms (sugar-free vs sugary). Additionally, these methods, generally include lemmatization as a preprocessing step, which loses morphological information (running vs run) and cannot capture out-of-vocabulary (OOV) words. Recent approaches [18, 28] learn a joint query-product matching model with character-trigram tokens (instead of lemmatized words) as inputs to deep learning encoders. The character trigrams allow morphological complexity and handle the OOV words [4] while the deep learning encoders capture
To alleviate the above challenges, we propose AtteNTive Hyperbolic (ANTHEM), illustrated in Figure 2, a novel attentive joint-learning framework that learns spatially-aware query representations. The representations capture the query’s hierarchical relations, and learns geometric operations (union and intersection) to match them with products. In addition to this, we provide an explainability mechanism that allows researchers to understand the ANTHEM’s internals and provide explainable search results.

Figure 2: Overview of the proposed ANTHEM model. The query entities are encoded in the hyperbolic space as hyperboloids. Attention over the individual entities and their intersections results in the final search space which is matched against the product embeddings.

In most of the industrial e-commerce settings, one would observe that products invariably lie in a hierarchy (see Figure 3) and relations between them are either hierarchical (nike shoes ⊂ shoes) or independent (nike ∩ adidas = ∅). Thus, unlike current approaches, we aim to preserve the hierarchical relations in addition to the semantic features. Hyperbolic spaces have proven to be more effective than Euclidean spaces at modeling such hierarchical relations [7, 11, 34]. Thus, we learn representations of our products as vectors in a hyperbolic space. Consequently, we also need to learn our query representations in the hyperbolic space. However, the query’s search space varies based on its broadness and relies on hierarchical relationships between its entities. To handle the query broadness, ANTHEM models the queries as hyperboloids with two hyperbolic vectors; the center and limit. The center defines the location of a query hyperboloid in the hyperbolic semantic space and its limit determines the search space (or volume) around the center. Thus, ANTHEM is capable of learning variable search volumes depending on the scope of a query. ANTHEM applies attention over individual tokens and their intersections to capture the significance of hierarchical relations, respectively, to the final search results in a hyperbolic Poincaré ball of unit radius. The activation units of the attention layers help analyze and explain our model’s search results for a query, thus making ANTHEM more interpretable compared to other methods. To the best of our knowledge, there is no existing work that models spatially-aware queries or utilizes attention in hyperbolic spaces to capture hierarchical relations.

Through a variety of experiments, we show that ANTHEM outperforms state-of-the-art baselines in product search, while being interpretable. To understand the semantic capabilities of ANTHEM’s query encoder in isolation, we test it on query-matching classification. In addition, we analyze the contribution of ANTHEM’s individual components to the overall performance through an ablation study. To summarize, the contributions of this paper include:

1. A novel product search framework, AtteNTive Hyperbolic Entity Model (ANTHEM) that utilizes token intersection/union and attention networks to compose queries as spatially-aware hyperboloids in a Poincaré ball, i.e., the query broadness is captured by the volume of hyperboloids.
2. A mechanism that utilizes attention units’ activation to understand the internal working of ANTHEM and explain its product search mechanism on sample queries.
3. Analysis of ANTHEM’s isolated query encoder and its ability to capture significant semantic features through the task of query matching on a popular e-commerce website.
4. An extensive set of empirical evaluation to study the performance of ANTHEM as a product search engine on a real-world consumer behavior dataset retrieved from a popular e-commerce website against state-of-the-art baselines.

The rest of this paper is organized as follows: Section 2 discusses the relevant background. Section 3 formulates the product search problem and explains the proposed ANTHEM framework. In Section 4, we describe the real-world e-commerce datasets, state-of-the-art baselines and evaluation metrics used to evaluate the proposed model. Section 5 concludes the paper.
2 RELATED WORK

In this section, we review various product search and semantic matching methods studied in the literature. We will also discuss several existing techniques that are developed for hyperbolic spaces.

2.1 Product Search

Product search algorithms are primarily motivated by existing works on search engines in the fields of Information Retrieval (IR) and Natural Language Processing (NLP), where the goal is to learn semantic information from queries and documents. However, product search mainly differs from traditional search in two key aspects; (i) product titles tend to be shorter than documents and (ii) signals (i.e., purchase information) are sparser than click-through data. Traditional approaches [2, 41] rely on lexical information to construct inverted indices and match queries with product titles. However, these methods do not consider semantic information which is imperative to handle synonymy and hypernymy [32]. DESM model [26] successfully leverages Word2Vec [22] vectors to rank documents for web search. Furthermore, Diaz et al. [9] expand the queries with gyrovector operations that are necessary for modeling network architectures. Gyrovector operations for Poincaré ball of radius c are Riemannian metric (g_2^E), Möbius addition (@_c), Möbius subtraction (⊖), exponential map (exp^E), and Möbius scalar product (∇c).

\[
g_c^X := \lambda_c^2 g^B \quad \text{where} \quad \lambda_c := \frac{2}{1 - ||x||^2} \quad (1)
\]

\[
x \oplus_c y := \frac{(1 + 2c(x, y) + c||y||^2)x + (1 - c||x||^2)y}{1 + 2c(x, y) + c^2||x||^2||y||^2} \quad (2)
\]

\[\exp^E(v) := \tanh \left( \sqrt{\frac{\lambda_c^2 ||v||^2}{2}} \right) \quad (4)
\]

Here, \(g^B = I_n\) is the Euclidean identity metric tensor and \(||x||\) is the Euclidean norm of \(x\). \(\lambda_c\) is the conformal factor between the Euclidean and hyperbolic metric tensor. It is set to a conventional curvature of -1 [11].

Recently, hyperbolic spaces have been successfully leveraged to learn representations of graph networks [13] and knowledge graphs [5, 38]. Given the hierarchical nature of product catalogue, we extend hyperbolic functions to 2-dimensional geometries in ANTHEM to improve its performance for product search.

3 THE PROPOSED ANTHEM MODEL

In this section, we will first discuss the problem setup for product search and query-matching, and then describe the different layers and overall architecture of ANTHEM (illustrated in Figure 4).

3.1 Problem Setup and Notations

3.1.1 Product Search. Given the query set \(Q\), for query \(q \in Q\), let \(\{s_1^q, s_2^q, ..., s_{|Q|}^q\} \in S^+\) and \(\{s_1^q, s_2^q, ..., s_{|Q|}^q\} \in S^-\) be the set of products with positive and negative purchase signal, respectively, i.e., for a given query, positive samples were the items purchased and negative samples were not purchased by the user from the given search results. The primary goal of product search is to recommend \(S^+\) for a query \(q\). To achieve this, we train ANTHEM to optimize a model \(P_\theta\) parameterized by \(\theta\) such that:

\[
\hat{y}_i^+ = P_\theta(s_i \in S^+|q_i, \theta), \quad \hat{y}_i^- = P_\theta(s_i \in S^-|q_i, \theta), \quad \hat{y}_i^+ + \hat{y}_i^- = 1
\]

\[
\theta = \arg \min_{\theta} \left( -\frac{1}{|S^+|} \sum_{i=1}^{|S^+|} y_i \log (\hat{y}_i^+) - \frac{1}{|S^-|} \sum_{i=1}^{|S^-|} (1 - y_i) \log (\hat{y}_i^-) \right)
\]

where for a given query \(q_i\), the probability of \(s_i\) being purchased and not purchased is provided by ANTHEM as \(\hat{y}_i^+\) and \(\hat{y}_i^-\), respectively. \(y_i\) denotes the Boolean ground truth purchase signal which is equal to 1, if product \(s_i\) is purchased for query \(q_i\) and 0, otherwise.

3.1.2 Query Matching. Due to the computational intensity of product search, most systems maintain pre-processed results for frequent queries [2]. Hence, it is more efficient to match a new query to an existing query result. In this problem, we aim to formulate query-matching as a multi-class classification task where the class labels define the similarity between query-pairs. Let \(D = \{(q_i, q_j, y_{ij})\}\) be the training dataset, where \(q_i\) and \(q_j\) are queries and \(y_{ij} \in Y\) is a multi-class categorical label denoting the relation between the queries; for example, query-pairs \((q_a, q_b), (q_a, q_c)\) and \((q_a, q_d)\) have labels \(y_{ab} = \text{class1}, y_{ac} = \text{class2}\) and \(y_{ad} = \text{class3}\).

The primary goal of query-matching is to predict class \(y_{ij}\) for a query-pair \((q_i, q_j)\). To this end, ANTHEM applies the query encoder
to optimize a model $X_\phi$ with parameters $\phi$ such that:

$$
w_{yc} = \frac{|\{q_i, q_j, y_j\} \in D \colon y_j = y_c|}{|D|}
$$

$$
\phi = \text{arg min}_\phi \left( \sum_{i=1}^{\left|D\right|} \sum_{j=1}^{\left|Y\right|} w_{y_i} y_{ij} \log \left( P_\phi \left( y_{ij} \mid q_i, q_j, \phi \right) \right) \right)
$$

where $w_{yc}$ is the sampling weight of class $y_c$ and $|\cdot|$ denotes the number of elements in the set.

The queries $q \in Q$ and products $s \in S$ are natural language text encoded as m-hot sparse character trigrams ($q, s \in \mathbb{R}^{|V|}$, $|V| = 48,807$). Applying them directly significantly increases the number of parameters, hence, we utilize an embedding layer ($\text{Embedding} : \mathbb{R}^{|V|} \rightarrow \mathbb{R}^d$, $d$ is empirically set to 128) to compress the raw embeddings and represent the semantic position of character-trigram $q_i \in q$ as center ($cen(q_i) \in \mathbb{R}^d$). Additionally, we add a limit parameter ($lim(q_i) \in \mathbb{R}^d$) initialized at zero to capture the trigram’s spatial awareness. Hence, the final Euclidean box embedding $B_{qi}$ is characterized by ($cen(q_i), lim(q_i)) \in \mathbb{R}^{2d}$:

$$
B_{qi} \equiv \left\{ v \in \mathbb{R}^d : cen(q_i) - lim(q_i) \leq v \leq cen(q_i) + lim(q_i) \right\}
$$

### 3.2 Layers of the ANTHEM

#### 3.2.1 Hyperbolic Transformation Layer

The embedding layers learn representations in the Euclidean spaces. In order to model the hierarchical relationships between products more effectively, we transform the embeddings to a hyperbolic space. The transformation from Euclidean ($\mathbb{R}^n$, $g^2$) to hyperbolic Poincaré ball ($\mathbb{H}^n$, $g^2$) is an $O(1)$ operation defined by the Riemannian manifold $\mathbb{H}^n = \{ x \in \mathbb{R}^n : ||x|| < 1 \}$ and metric, $g^2$ as given in Eq. (1). The Euclidean space encoded as m-hot sparse character trigrams ($q, s \in \mathbb{R}^{|V|}$, $|V| = 48,807$) is the total number of character trigrams with English characters and numbers.
queries \( q_{ij} = \{q_i, q_j\} \) as \( H_{q_{ij}} \):

\[
H_{q_{ij}} = \left( \text{cen} \left( H_{q_{ij}} \right), \lim \left( H_{q_{ij}} \right) \right)
\]

where \( \text{cen} \) is the Möbius scalar product, \( f(.) : \mathbb{H}^d \rightarrow \mathbb{H}^d \) is the multilayer perceptron (MLP), \( \sigma(.) \) is the sigmoid function, \( \text{Min}(.) \) and \( \text{exp}^c(.) \) are the element-wise minimum and Möbius exponentiation functions, respectively. The new intersection center and limit (shown in Figure 5) are aggregated by an attention layer [35] over the centers and shrinking with a minimum over queries. The main intuition here is that the semantic position (or center (cen)) of a set’s intersection is the weighted sum of its entities and the space boundary (or) limit (lim) is the least of all the entities’ boundaries.

### 3.3 Query Search Model

For a query containing \( n \) character trigrams, the individual query entity embeddings \( H_q = \{H_1, H_2, ..., H_n\} \in \mathbb{H}^{2d} \) and intersection embeddings \( H_{q_c} = \{H_1, H_2, ..., H_{an}\} \in \mathbb{H}^{2d} \) do not contribute equally to the final search hyperboloid. We capture this varying significance to scale the embeddings using self-attention networks [35]. The hyperboloid entity \( H_i \in H_q \cup H_{q_c} \) is scaled to \( e_i \in \mathbb{H}^d \) with function \( f_{att} \):

\[
f_{att}(H_i) = e_i = \frac{\sum \text{exp}^c(a_{ij})H_j}{\sqrt{d}}
\]

Given a query \( q \) containing \( m \) character-trigram entities, the query encoder returns a search space \( QS(q) \) which is a set of \( m \) and \( m^2 \) attention-scaled single and intersection hyperboloids, i.e., \( QS(q) = e_1, e_2, ..., e_{m(m+1)} \). Product \( s \) is encoded with a traditional attention network in the hyperbolic space [13] (shown in Figure 4). For a product \( s \), the product encoder returns a hyperbolic vector \( s \in \mathbb{H}^d \). Thus, unlike traditional approaches [18, 28], our query encoding is a set of hyperboloids and the product encoding is a hyperbolic vector. Thus, we need a specialized loss function that checks if the product vector is inside the query hyperboloids. The loss function \( L \) needs to capture the distance between \( s \) and \( QS(q) \), which is simply the distance between \( s \) and the nearest hyperboloid in \( QS \). Hence, we design our distance function as:

\[
dist(s, QS(q)) = \min \left\{ \left\{ d_{hyp}(H_i, s) \right\} : \forall H_i \in QS(q) \right\}
\]

\[
d_{hyp}(s, H_i) = d_{out}(s, H_i) \circ \gamma \left( d_{in}(s, H_i) \right)
\]

\[
d_{out}(s, H_i) = \left\| \text{Max}(d_{\ell}(s, H_{\text{max}}), 0) + \text{Max}(d_{\ell}(H_{\text{min}}, s), 0) \right\|_1
\]

\[
d_{in}(s, H_i) = \left\| \text{cen}(H_i) \circ \text{Min}(H_{\text{max}}, \text{Max}(H_{\text{min}}, s)) \right\|_1
\]

\[
d_{\ell}(x, y) = \cosh^{-1}\left( 1 + \frac{2}{\left\| x - y \right\| \left( 1 - \left\| x \right\| \left( 1 - \left\| y \right\| \right) \right)} \right)
\]

where \( d_{hyp}(x, y) \) is the hyperbolic distance between hyperbolic vectors \( x \) and \( y \), \( d_{in} \) and \( d_{out} \) is the distance of vector from the center and limits of the hyperboloid, respectively. \( y \) is a scalar weight given to \( d_{hyp} \). \( y = 0 \) implies a hard hyperboloid limit border and all vectors are either considered inside or outside, whereas, \( y = 1 \) implies no hyperboloid limit, thus, \( d_{hyp} \) is the distance between hyperboloid’s center and a product vector. For our experiments, we set \( y \) to 0.5. Choudhary et. al. [7] show that the super-linear nature of \( d_{hyp} \) increases density of entity clusters as we move down the hierarchy of a dataset, thus, increasing distance between different entities at the same level and decreasing distance between entities with the same parent. To convert the distance function’s output to a probabilistic distribution, we use a softmax layer [12]. Additionally, the loss function needs to minimize the distance between \( QS \) and \( s \in S^+ \) and maximize it between \( QS \) and \( s \in S^- \). Thus, the final loss for product \( S = S^+ \cup S^- \) is a cross-entropy function calculated as follows:

\[
y = \gamma \left( \sum_{q \in D} \log \left( \frac{\exp^c(dist(s, QS))}{\sum_{s \in S} \exp^c(dist(s, QS))} \right) \right)
\]

\[
L(y, y) = \left( \frac{1}{n} \sum_{i=1}^{n} y_i \log (\hat{y}_i) - (1 - y_i) \log (1 - \hat{y}_i) \right)
\]

### 3.4 Implementation Details

We implemented ANTHEM using Keras [6] on eight Nvidia V100 GPUs. For gradient descent in hyperbolic space, we adopt Riemannian Adam [3] with an initial learning rate of 0.0001 and standard \( \beta \) values of 0.9 and 0.999 and apply ReLU activation function [25]. The sensitivity of ANTHEM to hyper-parameters is presented in
Appendix D. For our empirical studies, we learn hyperboloid embeddings of $(2 \times 128)$ dimensions (d=128). The maximum sequence length of character trigram entities from queries and products is set to 28 and 128, respectively. This is set to fit the model’s parameters in our GPUs. The sequence limit of 28 completely covers $\approx 94\%$ of the queries in our datasets. Algorithm 1 provides the pseudo-code for training ANTHEM\textsuperscript{2} on the task of product search\textsuperscript{3}.

4 EXPERIMENTAL SETUP

This section will provide various experimental studies that analyze ANTHEM’s performance and compare with existing state-of-the-art baseline methods. We aim to answer the following research questions (RQs) in this paper.

- **RQ1:** Are the embeddings from proposed ANTHEM model better compared to those from state-of-the-art baseline methods?
- **RQ2:** Does ANTHEM’s query encoder capture semantic and hierarchical features for query-matching task?
- **RQ3:** What is the contribution of individual components to the overall performance of ANTHEM?
- **RQ4:** Can we explain the search results produced by ANTHEM?

Table 1: Basic statistics of the datasets used in experiments.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Class</th>
<th>Train</th>
<th>Valid</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-commerce</td>
<td># Positive</td>
<td>742,500</td>
<td>106,071</td>
<td>212,144</td>
</tr>
<tr>
<td>Product Search</td>
<td># Negative</td>
<td>3,861,002</td>
<td>551,571</td>
<td>9,582</td>
</tr>
<tr>
<td>E-commerce</td>
<td># Exact</td>
<td>456,652</td>
<td>65,236</td>
<td>1,271</td>
</tr>
<tr>
<td>Query Matching</td>
<td># Substitute</td>
<td>41,372</td>
<td>5,962</td>
<td>12,284</td>
</tr>
<tr>
<td>E-commerce</td>
<td># Complement</td>
<td>4,483</td>
<td>651</td>
<td>1,271</td>
</tr>
<tr>
<td></td>
<td># Irrelevant</td>
<td>22,865</td>
<td>3,266</td>
<td>6,714</td>
</tr>
<tr>
<td>Public E-commerce Search Relevance</td>
<td># Positive</td>
<td>4,786</td>
<td>684</td>
<td>1,367</td>
</tr>
<tr>
<td></td>
<td># Negative</td>
<td>9,582</td>
<td>1,369</td>
<td>2,738</td>
</tr>
</tbody>
</table>

4.1 Datasets

We conducted experiments on various product search and query-matching datasets. Table 1 presents additional details about the data distribution for these datasets.

- **E-commerce Product Search\textsuperscript{4}:** This dataset consists of 6.6M query-product pairs (retrieved from a popular e-commerce website) with a purchase signal which is a Boolean indicator that denotes whether a product was purchased\textsuperscript{5} or never purchased for a given query. The dataset is completely anonymized, and subsampled to enable efficient model training.

- **Public E-commerce Search Relevance\textsuperscript{6}:** This publicly available dataset consists of 20K labeled query-product pairs (columns query, product_title) with a purchase signal collected from the following five e-commerce websites: eBay, OverStock, Shop.com, Target, and Walmart. The purchase signal is a Boolean indicator based on the column relevance that denotes if a product is relevant if relevance > 0.5, else the product is considered irrelevant.

- **E-commerce Query Matching\textsuperscript{7}:** This dataset consists of 750K query-query pairs $(q_i, q_j)$ with a matching class, retrieved from a popular e-commerce website. The matching class will be one of the following four classes: (i) Exact: $q_i$ and $q_j$ are exact matches and produce the same results, e.g., playstation games and playstation games. (ii) Substitute: Parts of $q_i$ and $q_j$ can be substituted with one another, e.g., nike shoes and adidas shoes. (iii) Complement: $q_i$ and $q_j$ are complementary in meaning, e.g., phones and phone screen protectors. (iv) Unrelated: $q_i$ and $q_j$ are completely unrelated to each other, e.g., phones and shoes.

4.2 Baselines

To compare ANTHEM against the state-of-the-art frameworks, we select the following baselines based on previous research.

- **ARC-II** [17] utilizes a joint learning Siamese convolutional network to semantically match natural language sentences.
- **KNRM** [39] is another neural ranking model that uses a kernel pooling over cosine similarity between the query and document, followed by a dense layer to compute probabilistic scores.
- **DRMM** [14] is a neural ranking model that uses a histogram-like interaction vector to bin cosine similarity between the query and document into predefined intervals, followed by a dense layer to compute probabilistic scores.
- **aNMM** [40], similar to DRMM, computes a fixed-dimensional interaction vector by binning the cosine similarity between each of the query and document words. However, this model uses the total sum of the similarity between those word pairs as the features instead of using the counts of word-pairs.
- **MatchPyramid** [30] computes an interaction matrix between queries and document, and then passes it through CNN layers with dynamic pooling to compute sentence similarity.
- **C-DSSM** [32] is a twin-tower architecture that utilizes convolution networks to capture sequential information from character trigrams as inputs. This is currently the most scalable framework and applied in most of the product matching systems [28].
- **DUET** [23] combines the semantic and lexical matching strengths of C-DSSM and DRMM, in a deep convolutional architecture, to compute sentence similarity.
- **MV-LSTM** [36] employs multiple positional sentence representations to match sentences. The architecture aggregates an interaction matrix between different Bi-LSTM encoded positional sentence representations through multi-layer perceptrons.
- **BERT** [8] utilizes transformers to capture the co-dependence of different sentence units as attention weights. For this, BERT trains a language model by masking certain inputs. We adopt the large pre-trained BERT model and fine-tune it for our experiments.

The baselines are implemented in the Matchzoo framework [15] and the hyper-parameters are tuned using grid-search.

4.3 RQ1: Performance on Product Search

To analyze the efficacy of the query representations obtained from ANTHEM’s query encoder, we compare it against the state-of-the-art baselines on different product search datasets. ANTHEM takes a query-product pair as input and outputs the probability that the product belongs to the query’s search space ($P(s|q)$). The probability is calculated for $s \in S$ and the results are ranked to get the final search results. We evaluate our model using 5-fold cross validation on the following standard ranking metrics: Normalized Discounted Cumulative Gain (NDCG@K), Mean Average Precision (MAP),

\textsuperscript{2}Our code: https://github.com/amazon-research/hyperbolic-embeddings
\textsuperscript{3}Algorithm for query matching task is provided in Appendix C.
\textsuperscript{4}Proprietary dataset
\textsuperscript{5}To avoid anomalies, we only consider products purchased more than a predefined number of times.
\textsuperscript{6}https://data.world/crowdflower/ecommerce-search-relevance
Table 2: Performance comparison of the proposed ANTHEM architecture with several state-of-the-art baselines across proprietary and public product search datasets and evaluation metrics. E-ANTHEM is the Euclidean variant of ANTHEM without the Hyperbolic transformation layer. The results presented for the proprietary datasets are relative (in %) to the baseline ARC-II model. Exact evaluation results are presented for the public datasets. The best and second best results are highlighted in bold and underline, respectively. The symbol * indicates statistically significant improvement over BERT with a p-value ≤ 0.05.

<table>
<thead>
<tr>
<th>Models</th>
<th>E-commerce Product Search (in %)</th>
<th>Public E-commerce Search Relevance (in %)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NDCG@3</td>
<td>NDCG@5</td>
</tr>
<tr>
<td>ARC-II</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>KNNM</td>
<td>12.5</td>
<td>12.8</td>
</tr>
<tr>
<td>Duet</td>
<td>13.1</td>
<td>13.3</td>
</tr>
<tr>
<td>DRMM</td>
<td>20.5</td>
<td>22.8</td>
</tr>
<tr>
<td>aNNM</td>
<td>21.0</td>
<td>23.9</td>
</tr>
<tr>
<td>MatchPyramid</td>
<td>25.6</td>
<td>25.6</td>
</tr>
<tr>
<td>C-DSSM</td>
<td>31.3</td>
<td>29.4</td>
</tr>
<tr>
<td>MV-LSTM</td>
<td>34.7</td>
<td>33.3</td>
</tr>
<tr>
<td>BERT</td>
<td>38.6</td>
<td>37.2</td>
</tr>
<tr>
<td>E-ANTHEM</td>
<td>49.4</td>
<td>46.7</td>
</tr>
<tr>
<td>ANTHEM</td>
<td>51.4*</td>
<td>48.9*</td>
</tr>
</tbody>
</table>

Figure 6: Example results for sample queries shown by our model and the best performing baseline. The results given are the third result for the query. The first two results are not shown here because they were equally appropriate for the query and the figure aims to show the differences between BERT and our model.

and Mean Reciprocal Rank (MRR). The datasets are split into training, valid and test sets of ratio 7:1:2, as given in Table 1. The results on the test set are presented in Table 2.

From the results, we observe that ANTHEM outperforms the state-of-the-art baselines across datasets by 10% – 15% in all the evaluation metrics. Additionally, we can notice that utilizing Euclidean spaces also improves the performance by ≈ 9%. This is empirical evidence that ANTHEM’s spatially-aware query hyperboloids form better search space for E-commerce queries. Another point of note is that BERT (the best-performing baseline) has over 100M parameters, whereas, ANTHEM is able to achieve better results with only 6.37M parameters. In addition, we quantitatively analyze the results of our model and the state-of-the-art baselines. From the sample, shown in Figure 6, we observe that BERT is not able to capture the correct significance of brand intent (“aveno” in query and “aveeno” in products) from the query and emphasizes on the actual product type. ANTHEM, on the other hand, is able to provide more appropriate results due to the use of character-trigrams and inter-entity relations that are able to infer the importance of brand information and relation between “aveno” and “moisturizer”, respectively. In the second case, for the query “pokemon movie”, BERT focuses on the brand information and cannot differentiate the product type from the given title, thus, recommending an unsuitable product, whereas, the inter-entity relation in ANTHEM define the query as an intersection between “pokemon” and “movie” entities, consequently, providing more appropriate results. In the last case, “playstation 4” is a short query with an ambiguous user intent. However, we notice that our model leverages hierarchical brand information to select the more pertinent product, whereas, BERT returns an improper result due to the ambiguity. This demonstrates the importance of hierarchical information in product search.

4.4 RQ2: Performance on Query Matching

To analyze the efficiency of our model’s query encoder in isolation, we compare it against the state-of-the-art baselines on the query matching dataset. In this experiment, we pair the query encoder part of ANTHEM with a matching function (cosine similarity) in a siamese learning framework. The architecture takes a query pair as input and outputs the similarity between the queries. We evaluate our model with 5-fold cross validation on the standard classification metrics: Accuracy, F-score, and Area under ROC (AUC). The datasets are split into training, valid and test sets of ratio 7:1:2, as given in Table 1. The results are presented in Table 4.

From the results, we observe that ANTHEM is able to outperform the state-of-the-art baselines across datasets by 4% – 8% in the evaluation metrics. E-ANTHEM, utilizing Euclidean spaces also improves the performance by 3% – 6%. This is empirical evidence that ANTHEM’s query encoder is able to capture the most significant semantic features.

4.5 RQ3: Ablation Study

In this experiment, we study the importance of different components that contribute to the overall performance of our proposed model. The components studied in our ablation experiments are:
(a) Mapping brands to products  
(b) Translating semantically similar phrases  
(c) Semantic/lexical matching

Figure 7: Explainability Study. Significance of query entities (y-axis) to the product entities (x-axis) analyzed through the final attention layer of the ANTHEM product search model. (a) ANTHEM is able to learn a matching from brand Tony Perroti to item tokens leather and briefcase, which enables better query-product matching. (b) ANTHEM is able to semantically map query term daily moisturizer to a lexically different term in the product lotion. (c) ANTHEM is able to leverage hierarchical brand from products Wilson SGX and match to queries with no direct semantic/lexical similarity golf clubs.

Table 5: Performance comparison of the contributions from different components: Hyperbolic layer (H), Intersection layer (I), and Limit parameter (L). The results presented for the proprietary dataset are relative to the performance of first row (w/o L w/o I w/o H). ‘w/o’ stands for without. For public datasets, exact evaluation metrics are presented.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>E-commerce Product Search (in %)</th>
<th>Public E-commerce Search Relevance (in %)</th>
<th>E-commerce Query Matching (in %)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NDCG@3</td>
<td>NDCG@5</td>
<td>NDCG@10</td>
</tr>
<tr>
<td>w/o L w/o I w/o H</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>w/o L w/o I</td>
<td>6.4</td>
<td>6.2</td>
<td>7.5</td>
</tr>
<tr>
<td>w/o I</td>
<td>23.4</td>
<td>22.9</td>
<td>36.8</td>
</tr>
<tr>
<td>ANTHEM</td>
<td>41.5</td>
<td>39.6</td>
<td>67.3</td>
</tr>
</tbody>
</table>

Table 6: Performance comparison of the proposed ANTHEM model with several state-of-the-art baselines on the E-commerce query matching dataset and evaluation metrics. The results presented are relative to the baseline ARC-II.

<table>
<thead>
<tr>
<th>Models</th>
<th>Accuracy (in %)</th>
<th>F-score (in %)</th>
<th>AUC (in %)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARC-II</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>KNRM</td>
<td>-4.1</td>
<td>-24.9</td>
<td>-19.1</td>
</tr>
<tr>
<td>DRMM</td>
<td>25.1</td>
<td>15.4</td>
<td>33.1</td>
</tr>
<tr>
<td>aNMM</td>
<td>-1.3</td>
<td>-8.6</td>
<td>4.0</td>
</tr>
<tr>
<td>MatchPyramid</td>
<td>-14.5</td>
<td>-17.8</td>
<td>-9.7</td>
</tr>
<tr>
<td>C-DSSM</td>
<td>21.2</td>
<td>21.7</td>
<td>30.1</td>
</tr>
<tr>
<td>DUET</td>
<td>-2.3</td>
<td>-4.7</td>
<td>-6.9</td>
</tr>
<tr>
<td>MV-LSTM</td>
<td>71.1</td>
<td>21.2</td>
<td>48.9</td>
</tr>
<tr>
<td>BERT</td>
<td>40.1</td>
<td>33.5</td>
<td>54.7</td>
</tr>
<tr>
<td>E-ANTHEM</td>
<td>43.2</td>
<td>40.3</td>
<td>61.4</td>
</tr>
<tr>
<td>ANTHEM</td>
<td>43.9</td>
<td>40.8</td>
<td>62.6</td>
</tr>
</tbody>
</table>

(i) Hyperbolic layer (hierarchical features), (ii) Intersection layer (capturing inter-entity relations) and (iii) Limit parameter (spatial-awareness). The results of our experiments are presented in Table 3. The results show that the Intersection layer and Limit parameter contribute ≈ 25% to the overall performance of ANTHEM. Thus, we conclude that capturing inter-entity relationships in spatially-aware representations aid the performance of product search. Furthermore, removing the Hyperbolic layer decreases the performance by an additional 3%–8% which shows the contribution of hierarchical information to the overall performance.

4.6 RQ4: Explainability Study

In this section, we analyze the internal working and significance of the query to the final results by utilizing the activation units of ANTHEM’s attention layers. The attention units of a few sample queries are depicted in Figure 7 which provide a mechanism for researchers to understand the internal functions of our model. ANTHEM is able to match brands to products (Fig. 7a), translate semantically similar phrases (Fig. 7b) and leverage hierarchical information for semantic/lexical query-product matching (Fig. 7c). Thus, we conclude that concurrently utilizing product’s hierarchical information (as Hyperboloids) and inter-product relation (as intersection) leads to better product representations. Also, such insights allow other researchers to independently analyze our model’s suitability in their own applications and facilitate its integration.

5 CONCLUSION

In this paper, we presented ANTHEM, a novel product search framework that utilizes inter-token intersection/union and attention networks to encode query search spaces as spatially-aware hyperboloids in a Poincaré ball. We emphasized the utility of leveraging hierarchical information in product search and the need for spatially-aware query representations in the e-commerce domain. We performed an extensive set of empirical evaluation to study the performance and interpretability of our model as a product search engine on real-world query data collected from a popular e-commerce website. Finally, we validated the capability of our isolated query encoder in a query-matching task and analyzed the contribution of its components through an ablation study. Additionally, given the multitude of industrial applications, we also provide an explainability mechanism for researchers to analyze and integrate ANTHEM in their own architectures. We hope to provide a new perspective towards composing e-commerce queries as intersection and union of their individual tokens, instead of processing them as a sequence of tokens.
REFERENCES
A  BROADER IMPACT
ANTHEM has the potential to have a large impact on product search and discovery. A vast majority of customers across countries start their shopping journey on e-commerce websites via a search functionality. Given the several millions of customers who interact with these systems, any improvements in performance of these systems (however small they are) has a large impact on the user base. Our work is aimed at practitioners and researchers in the broader data mining and machine learning communities who work in the domain of representation learning, particularly learning in the presence of hierarchical information.

Current systems model customer behaviors to provide more contextual information to improve search results. However, this customer information is both sensitive in nature and also a substantial source of bias [19]. In ANTHEM, we aim to provide a possible alternative which considers possible intents and statistically infers the right intent through historical purchases. Additionally, we design our model in a joint learning framework so that it conforms to existing architectures for easier deployment and is applicable to additional problems such as web search and semantic matching.

B  COMPUTATIONAL COMPLEXITY
To analyze ANTHEM’s potential of deployment in an industrial setting, we need to study the computational both in terms of its training and inference times. The model’s parameter study, training time and inference run-time are provided in Table 5. To maintain a fair comparison, we do not include the overheads involved in the inference process such as loading the model and the request processing time of servers. We observe that the run-time of our models ANTHEM and E-ANTHEM are slightly higher than previous methods. The reason for this is the use of intersection and union which are quadratic operations. However, we note that length(query) << length(answers) in product search engines, thus, the added complexity does not affect the runtime significantly. The difference in runtime is ~ 10 seconds for 657K validation samples. Additionally, we also report a much lower number of model parameters in our models compared to the best performing baseline, i.e., BERT. This implies a lower training period (an advantage of our models ANTHEM and E-ANTHEM are slightly higher than previous source of bias [19]. In ANTHEM, we aim to provide a possible alternative which considers possible intents and statistically infers the right intent through historical purchases. Additionally, we design our model in a joint learning framework so that it conforms to existing architectures for easier deployment and is applicable to additional problems such as web search and semantic matching.

C  QUERY-MATCHING ALGORITHM
Algorithm 2 provides the pseudo-code for training ANTHEM for the task of query matching. The algorithm utilizes cosine similarity to match the queries and return a probabilistic similarity measure for optimization (gradient back-propagation) using cross-entropy loss. The cosine similarity between query representations (set of hyperboloids) $Q_Sq$ and $QS_r$ is calculated as:

$$\hat{y} = \mu \left( \frac{\|Q_Sq \cdot Q_Sr\|}{\|Q_Sq\| \cdot \|Q_Sr\|} \right) \quad \forall Q_Sq, Q_Sr \in QSr$$

(14)

where $\mu(.)$ and $\|\|$ represent the mean and Euclidean norm functions, respectively.

D  SENSITIVITY TO HYPER-PARAMETERS
In this section, we study the sensitivity of our model with respect to the hyper-parameters. First, we analyze the convergence of our model across epochs and then we proceed to analyze the loss with
varying dropout rates and lengths of query/product titles. Here, the length refers to number of character trigrams in the query or product. The results are presented in Figure 8.

In Figure 8a, we observe that ANTHEM is able to converge in under 50 epochs with Riemannian Adam optimizer (details provided in Section 3.4). Figure 8b illustrates the advantages of using dropout for efficient convergence and avoid overfitting. We observe that a dropout rate of 0.5 results in the most optimal solution and, hence, we adopt that in ANTHEM. In addition to this, we also need to find the most optimal query and product length. The model’s complexity directly depends on these lengths. Hence, an optimal solution will significantly affect the training time. From Figure 8c, we observe that a query length of 32 with a product length of 128, provides the most optimal result. However, due to computational constraints, i.e., GPU VRAM < 8GB, we utilize a maximum query length of 28 and product length of 128. Finally, in the case of embedding units, we observe from Figure 8d that 128 is the optimal number of dimensions for least loss, and hence we utilize that for our model.

The final hyper-parameters adopted for E-ANTHEM and ANTHEM are query length of 28, product length of 512 and dropout rate of 0.5. For the baselines, the query length is 128 and product length is 512. The dropout rate for ARC-II, KNRM, DUET, DRMM, and aNMM is 0.2 and for MatchPyramid, C-DSSM, MV-LSTM, and BERT it is 0.5. The above hyper-parameters for the baselines are determined after extensive experimentation for the best performance.

E NUMBER OF GPUS

Figure 9 shows the dependence of training time on the number of GPUs, which depicts the feasibility of ANTHEM’s parallelization. We notice that ANTHEM has a lower training time than the best performing comparison baseline, BERT.