Towards a Generalized Similarity Service

Anuradha Bhamidipaty
IBM Research
Yorktown Heights, NY
anubham@us.ibm.com

Daniel Gruen
IBM Research
Cambridge, MA
daniel_gruen@us.ibm.com

Jeffrey O. Kephart
IBM Research
Yorktown Heights, NY
kephart@us.ibm.com

Siva Sankalp Patel
IBM Research
Yorktown Heights, NY

Justin Platz
IBM Research
Yorktown Heights, NY
jmplatz@us.ibm.com

Danny Soroker
IBM Research
Yorktown Heights, NY
soroker@us.ibm.com

John Vergo
IBM Research
Yorktown Heights, NY
jvergo@us.ibm.com

Alan Webb
IBM Research
Yorktown Heights, NY
alan_webb@us.ibm.com

ABSTRACT

Similarity is an integral part of learning, both human learning and machine learning. Similarity-driven reasoning, analogy, learning and explanation are critical for AI to become truly robust. The complexity in learning a measure of similarity particularly stems from the fact that there is no single notion of similarity for a set of objects. The definition of similarity is critically dependent on the impression of the user performing the task. To truly provide a similarity service, the system has to be able to learn the notion of similarity in real-time, interacting with the user. In this paper, we introduce a vision of generalized similarity service that attempts to learn an individual's similarity function. A conceptual framework describing the system capabilities for such a service is presented. Implementation of this framework is applied to the domain of company similarity. A preliminary user study highlights the importance of generalized similarity service.

CCS CONCEPTS

• **Information systems → Users and interactive retrieval**; *Similarity measures*; *Novelty in information retrieval*;

KEYWORDS

Similarity, user intent, user interaction

1 INTRODUCTION

Similarity is fundamental to learning. It is well established that reasoning and learning by analogy are important aspects of human cognition [9, 10]. In [26] the author emphasizes the importance of similarity for problem-solving and reasoning, and for AI systems of the future. In Machine learning, the problem of similarity has been addressed in the context of many applications such as those in information retrieval, recommendation systems, computer vision, and natural language processing. Most approaches employ distance metrics to approximate similarity between two objects (e.g., people, images, things etc.). Distance metric learning is accomplished with input from domain experts, or the presence of large amounts of data as well as ground truth of pairs of labeled instances. The distance metric thus learned is applied globally to all instances of dataset with the underlying assumption that similarity can be universally defined for all users of the dataset. But human perception of similarity is complex and often involves subjective judgment. In [11] the author refers to similarity as a phenomenon that is not unitary. Similarity not only varies with context, intentions, and characteristics of the user, but also by how it is calculated in diverse tasks.

Computing similarity is typically performed in the context of search and discovery tasks. We suggest that at the time they enter such an interaction with the system, users have a mental model of the entity they are seeking. However, the user's mental model is almost certain to be an approximation of what they are looking for. There are a number of reasons for this:

- Complex entity types are represented by 100's or 1000's of dimensions and the user is unlikely to care about all the attributes of an entity. Instead, a subset of the full representation of the entity will be "in scope" for a given context.
- Even if a user can accurately specify a subset of the attributes that they care about, it is likely they will weigh the importance of each attribute differently. It is non-trivial for a user to accurately articulate the relative weights of a significant number of attributes.
- The attributes of interest and their weights are likely to undergo changes as the search and discovery process unfolds. Consider a user may have an unrealistic expectation for individual attributes or combinations of attributes. The user may initially be interested in finding people who have both won the Noble Peace prize and won a gold medal in the Olympics 100M dash. Upon inspection of the search results, the user's constraints may be relaxed to something like "find me people who are smart and fast".

IDEA @ KDD'18, August 20th, 2018, London, United Kingdom

It is important to provide a user with a similarity service which is not constrained by one universally applicable distance metric for all objects in the dataset.

We define **Generalized Similarity** as a computing service, comprised of algorithms and tools, that is capable of finding entities similar to a reference entity (which may be real or idealized), or to a collection of such reference entities, according to a notion of similarity that may evolve over the course of a user's interaction with that service. The service is capable of interacting with users in natural, intuitive ways to capture and (at least indirectly) refine their notions of similarity. Typically, an entity will be represented as a collection of unstructured and structured attributes. Our vision is motivated by our work in the field of recommending companies as acquisition targets to the business development personnel of a large corporation. Some motivating scenarios are described in section 1.1.

Our contributions in this work-in-progress paper are:

(1) We introduce a vision of a generalized similarity service, and present the high-level capabilities that a similarity service should exhibit.
(2) We describe the application of the similarity service to the domain of company similarity.
(3) We present preliminary user study results.

Our work, to date, has been on a single type of entity, namely companies. Our pursuit of Generalized Similarity envisions an architecture that allows many of the core algorithms and tools to be reused an/or extended to support simple and rapid implementations of the similarity service for new entity types. A detailed discussion of the architecture is beyond the scope of this article.

1.1 Motivating Scenarios

Observations of business users in various roles as they compiled lists of companies makes clear the extent to which context influenced intent and thus which criteria users would use, often implicitly, to determine whether a candidate was "similar" to others on a list they were compiling. This context reflected their role and intended use, for example whether they were a business development practitioner searching for potential companies to acquire, a project manager searching for suppliers, or a salesperson looking for new customers.

In one commonly observed use case, the user begins with a small set of known companies — or perhaps even a single company — and looks for others like it. This use case arises in many different situations, leading to different similarity criteria and thus different judgments on whether a specific company belongs with others on their list. Context can influence the extent to which a specific feature is important or irrelevant, and, if relevant, if the user's goal in finding similar companies is to adhere to or expand from values for that feature currently on their list. Examples based our observations include:

(1) A user is only interested in companies based in Canada. **All her starting examples are in Canada, and she would like the others to be there as well.**
(2) A firm has partnered successfully with companies in several European countries, and hopes to expand by partnering in other geographies. **All their existing examples are in Europe. They want companies that are like those but *not* in Europe.**
(3) A user wants to build a comprehensive list of companies from around the world. He is familiar with the local market, so all his examples happen to be in it. **He'd like help filling in the list with companies from a broad range of countries, so diversity of geography is important to him.**
(4) **Geography is irrelevant to the user.** No attention should be paid to where the companies already on the list happen to be based.

Context can also influence whether the user intends that a specific feature simply be present in the description of a candidate, or predominant, as seen in the following examples:

(1) An executive wants to acquire a company that produces fitness tracking hardware. She would prefer one that focuses on this as a majority of their business; otherwise she would have to acquire business operations she doesn't want.
(2) A sales development practitioner is looking for customers for health-related components and services, and would therefore like to find as many fitness device makers as possible. He doesn't care if they have other product lines as well.
(3) A competitive analysis team wants to monitor what all the players in a market are doing. They want this list to be as exhaustive as possible so they don't miss anything.

In many cases, users can describe an ideal candidate. In such scenarios the user explicitly articulates a set of attributes that are used by the system to search for a set of similar companies. For example, the user may express interest in "AI acceleration hardware vendors who are headquartered in the European Union and who have between 100 and 300 employees". Even in such cases much is unstated; users we observed would typically not want a company currently strongly partnered with a competitor or built on technology incompatible with that used in their other products. Such criteria are often understood implicitly by colleagues, but must be revealed through interaction with the system, and potentially learned given sufficient examples of past lists compiled in similar situations. (Here too the question of similarity arises in determining what makes a situation similar to another.)

Note that even in situations in which users must ultimately decide on a single result, such as the one company to acquire or single city to choose as their headquarters, showing a list of candidates is necessary as an intermediate stage. Doing so enables them to understand the universe of possibilities, helps surface criteria they did not realize or remember were important, and can be crucial to increasing confidence in the selected result as the best of the available options.

2 USER NEEDS AND SYSTEM DESIGN

Our work has been informed throughout by direct involvement of expert business development practitioners in various divisions of a large corporation. This has included in-depth semi-structured interviews with six practitioners who described their current practices, walked through a typical project and described the materials they used and lists they were building and maintaining, described current exploration projects, and identified the main pain-points

they faced. In addition, we conducted a one-day workshop with ten practitioners to further probe practices and envision potential system features. Finally, a dedicated team of four business development professionals met with us on a regular basis to review designs and test prototypes as we iteratively developed the similarity tool.

Current methods for identifying companies include performing Internet searches, reading analyst reports, monitoring news, patent, and trade publications, and searching specialized company databases. Especially with established industry segments, articles identifying "top" companies often exist; while users found these useful as one source, they were rarely comprehensive and could not be relied upon alone. With emerging areas such as those in which startups are often engaged, or with niche areas involving very specific technologies or applications, no such lists of top companies typically exist. Users report spending hours, days and in some cases up to two weeks developing lists of relevant companies before even starting more in-depth analyses.

Learnings from these sessions with users indicated our service would need to be fast enough to aid real-time discovery, capable of integrating into existing workflows and robust enough to distinguish between different notions of similarity. Thus, the features, algorithms, and associated hyperparameters we developed for our service were driven by real user feedback and motivated by the use-case of discovering companies.

However, the system we developed served just one domain. In order to serve a multitude of domains, we formed two hypotheses which detail the design such a system would need.

First, a generalized similarity service would be required to maintain a notion of universal semantic types which could span across domains. In this way, an entity agnostic generalized similarity service could easily encompass a new domain with a basic ontological understanding of universal types. A company name, person name, and a song name all share a similar semantic value. Accordingly, comparisons between names in different domains, whether it be companies, people, or songs can utilize many of the same operations.

Second, a generalized similarity service would require domain-specific comparators tuned via comparison to user judgments. As an example, domain-specific comparators tuned by users would be required in order to compare "net-worth" for companies, people or songs. Streamlined user sessions, like we had with business users, could be used to extend domains by learning from users.

3 SYSTEM CAPABILITIES

Our research goal is the creation of a generalized similarity service. The service must include corpora, tools, algorithms, and interaction mechanisms necessary for users to search for, analyze, and reason about entities that are similar to one another, to a user's model of an entity, or a combination of the two. The service adapts to the intent and context of the user.

We define a set of capabilities that a Generalized Similarity service will exhibit, shown in Figure 1.

3.1 User Mental Model

In the context of generalized similarity, intent is a critical ingredient to finding similar entities. If a user says, "find me companies bigger than XYZ", the user's main intent here is to find companies along with an additional requirement that the companies be "bigger" than XYZ. If, however, the user says "find me employees working at XYZ who have the skill ABC", the intent is to find employees along with two additional constraints that the employees have the skill ABC and that their current employer be XYZ. Intents are global properties which highlight the goal of the user. Another task which is closely related to intent recognition is that of slot detection[3]. Slots are mentions of certain entities within the query, which together with intent can be used as parameters for similarity. For example, in the examples above, *XYZ* is a mention of entity type *Company*. Similarly, *ABC* would be a mention of *EmployeeSkill*.

Intent recognition is usually treated as a semantic utterance classification problem whereas slot detection is treated as a sequence labeling task. Popular classifiers for the former task include support vector machines (SVMs) [31] and deep neural network models [28]. For sequence labeling, popular approaches include conditional random fields (CRFs) [25] and maximum entropy Markov models (MEMMs)[20]. The current state-of-the-art approaches for these two tasks make use of Recurrent Neural Networks (RNNs) [17, 22, 34]. Joint approaches [12, 18, 21], which solve both the sequence tagging problem (slot detection) as well as the sentence classification (intent recognition) problem with a single model, have been proposed in recent literature which achieve state-of-the-art as well. Joint models make use of the encoder-decoder architecture originally proposed for machine translation [7].

Finally, we expect that research in search result diversification [13] will play an important role in understanding the user's intent. By systematically diversifying search results based on an intent model, it may be possible to infer users' intent based on their selection of results.

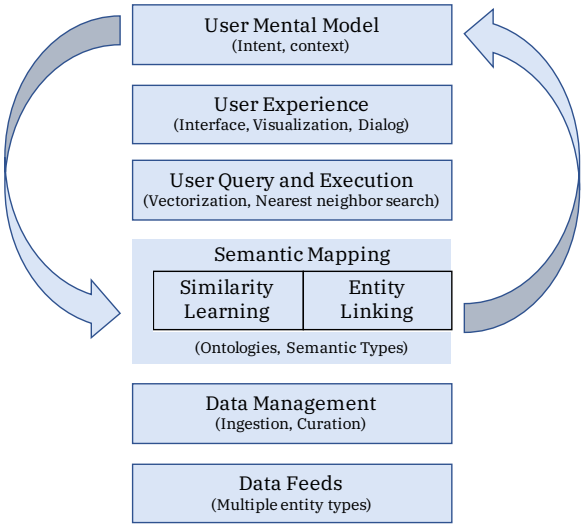

Figure 1: Conceptual Model of Generalized Similarity

3.2 User Experience

The interpretation of search results can benefit from novel presentation and interaction techniques, especially when such techniques

are designed to help users understand aspects of similarity. Our interest is in complex entity types which are typically characterized by high dimensionality. We have explored dimensionality reduction techniques such as t-SNE [19] and heat maps to aid users in gaining insight into search results. We believe the ability to interactively explore high dimensional spaces [5] is highly beneficial to a generalized similarity service.

Engaging in natural language dialog with a user will be important. Language is the natural basis for generalized communication and it is likely that in the future, systems able to communicate in this way will be distinguished not by any notion of user interface but by how well they act upon the user's intent. Since we are dealing with high dimensional data and complex user mental models, we posit that multi-modal [24] conversational interfaces [15] will ultimately provide the most natural and effective means of interaction for Generalized Similarity services.

3.3 User Query and Execution

Similarity search is focused on finding an instance of an entity that is close to a query entity[32]. Our use cases involve the discovery of a set of instances close to the query, hence techniques that identify a set of neighbors, i.e. approximate nearest neighbor (ANN) search algorithms, are central to generalized similarity. One relevant family of algorithms is Locality Sensitive Hashing (LSH). In LSH, hashes (codes) of entities are created with the express goal of maintaining relative distance in the input space in the hash codes themselves. Ideally, very close neighbors in the input space are hashed to the same code, making it very efficient to determine close neighbors. LSH was introduced in 1997 [4]. Since then there has been a steady progression of LSH algorithms (e.g. Simhash [6], Spectral hashing [33], Cover Trees [2], Product Quantization [14]). We are experimenting with different LSH algorithms to understand their efficacy with generalized similarity.

Capability to handle nearly all forms of information including unstructured information will be essential. Entities are ultimately considered to be a collection of attributes (features, dimensions) that are typically a combination of structured and unstructured data.

Each of the aforementioned hashing functions are predicated on being able to represent entities in a vector space. Since our interest is in highly complex entities (e.g. people and companies), the data that are associated with the entities occupy a wide range of representations including structured and unstructured data (text, image, video, etc). These data formats must be converted to a vector representation in order to be useful to existing ANN algorithms. Our current implementation focuses on text. We have implemented tf-idf [27] as our text vectorization technique. We have also explored the use of Word2Vec [23] but it did not achieve measurable performance improvements over tf-idf. Other techniques we anticipate using are one hot encoding (for categorical data) and Simhashing for nominal data. The vectorization methods we intend to explore in the future are likely to be most effective when they encode semantics of the entity attributes in a way that ANN algorithms can meaningfully discriminate among entities based on the relevance of the encoded semantics. A complete review of vectorization techniques for all modalities (e.g. text, image, video, sound, etc.) is beyond the scope of this paper. We anticipate a lot of experimentation to determine the best techniques for the various data types and use cases that are ultimately required for generalized similarity.

3.4 Semantic Mapping

Semantic mapping is an abstraction that allows us to operate across a broad range of real-world concepts without directly interacting with the raw data and their formats. The representation of 'real-world' data is manifold and its management is a long-standing challenge of software engineering. Semantic mapping allows us to utilize a standard representation for a particular meaning and transparently provide transformations of actual data to a normal form (without changing the sourced representation) at time of access. This capability focuses on the replacement of *data* with *meaning* at a foundational level, and in so doing propagates a natural semantic model that permeates the entire system; not just the user interface. This is not an entirely new idea; the work of the W3C Semantic Web project is directly concerned with this principle, and much work has already been done to establish thesauri (ISO 25964) that formalize this principle.

Not having semantic abstraction means that the user of the system is only able to express intent in ways allowed by the system's implementation. This has the disadvantage that it both reveals, and is dependent upon, the specific nature of the underlying system; making any attempt to evolve the system disruptive to the user, exposing the effects of intentional and unintentional obsolescence on existing results.

Semantic mapping also helps in analyzing collection of entities. We have observed use cases where users maintain list of entities, like companies that are under consideration for acquisition, partnership or as potential customers. In all such scenarios, it is useful to be able to understand what makes the companies on the list similar and use that knowledge to search for other companies that naturally belong on the list.

The ingestion of data from multiple data sources (see subsection on Data Sources) requires the linking of data to entity instances. We include entity resolution in the semantic mapping subsection because an understanding of the semantics of attributes greatly enhances our ability to accurately link attributes to an entity. The entity resolution problem is a challenging area of research in its own right. [29][8]

Semantic abstraction suggests a way that we can use natural language and ontologies to establish context, and guide interaction with the underlying information models, based upon a derived representation of the user's mental model and intended query. Not all of this needs to be implemented at once, and any implementation is expected to be gradual. In contrast, the principles must permeate both architecture and design from the start.

Ontologies are used as a way to represent entity types such as company, organization, or industry with each ontology containing zero or more elements representing the "ideas" that constitute the entity. For example, a company may be described using ideas of location, size, industry, funding, and so on.

From a practical perspective this ontological model fits conveniently with the principles of an object-relational model normally used to isolate a program from the physical realities of how data

is stored. We extend this paradigm to provide operations that use the semantic representation of the data with a set of contextually appropriate operations. Every semantic value is represented as an object and raw data is never accessed directly.

3.5 Data Management

Management of data in AI systems, including ingestion, curation, annotation and entity resolution, is known to present substantial challenges and account for much of the effort when developing new AI solutions.

Data sources can include public data, licensed data, data extracted from web crawls and many other sources. Many data are "noisy"(web, crowd sourced data, news, blog, etc.). It is also common to encounter conflicting data, which should be managed in ways that are common across entity types. For a generalized similarity service to provide valuable results, it is necessary to keep the information in the system "fresh". The data management subsystem must therefore be capable of efficiently and periodically refreshing the data from a disparate set of data sources.

3.6 Data Sources

We envision an engine that can work with multiple entity types (People, Companies, Charities, Schools, Countries, Houses, etc.). The engine's core components will be agnostic to any individual entity type. We do expect that some modification of the ontological model will be required for each new entity type, but the goal is to create an engine that minimizes the development cost of implementing a similarity capability for new entity types or use cases. The intent here is that semantic mapping be a continuum that connects the language of the request to the entities and operations used to satisfy it; an exciting possibility being that a simple restatement may be all that is required when intent is misunderstood. In rare cases specialization may require that new entities be introduced, but in general it is limited to new operations over existing entities. An example of where we have observed specialization in the company domain is when we pre-process descriptive data to remove "noisy" terms such as Corp. and Inc.

Our experience with building a similarity service for companies taught us that a robust representation of companies requires the integration of multiple data sources in order to provide coverage across the world of companies and deep knowledge of individual companies. Coverage refers to having as complete a set of entities as possible. When dealing with companies, we found that different data sources had strengths and weaknesses with respect to the completeness of their data stemming from geographic, industry, company type and other focii. Similarly, individual data sources tended to focus on different attributes of companies, e.g. some had relatively complete data on company financials and others on venture capital and/or private equity funding. No single data source contained all the data we required, necessitating integration and entity resolution of data from multiple sources.

3.7 Iteration and Learning

The Generalized Similarity use case described in Section 1.1, Motivating Use Cases, are fundamentally iterative in nature. In all cases, users refine their understanding of the entities they are working with, informed through exploration of the space. There are several opportunities to employ learning techniques as these iterations occur. There are opportunities to learn the users' intent, and to learn how other users with the same intent altered their understanding of what they were searching for. As an example, if a user rejects all suggestions of companies with fewer than 100 employees, the system can infer that their intent is to focus on larger options. This can be tested by probing with other suggestions, or by explicitly requesting confirmation. We also expect that different intents will result in weighing the importance of entity attributes, which can be learned through machine learning. This knowledge can then be applied to support new users with similar intents.

As is detailed in section 4.2, we also employed learning techniques to tune the similarity algorithm.

4 IMPLEMENTATION FOR COMPANY SIMILARITY

Our initial foray in building a system integrating similarity was in the context of a tool we built to assist business users with construction of lists of companies, and discovery of additional companies that would be good candidates for addition to their lists. We describe here the prototype system we built for them.

Users can create named lists, and manually add companies they already know about to their lists by searching for their names in a database of approximately 2.5 million known companies. The company corpus is an integration of two commercially available data sets (Crunchbase [1] and CapitalIQ [2]) and is being augmented with publically available data from other sources. By clicking on a search result, a user can see a summary of information about the company as shown in Figure 2. A user can choose to add the company to a list, or view more details about it. Searches are stored and can be later utilized throughout the system.

Once a user has added companies to a list, they can apply a match-multiple algorithm (section 4.4) to find companies similar to the list overall. After selecting the "Similar Companies" tab, a user will be presented with suggested companies. By default, suggested companies are shown in a matrix in descending order with the most similar on top as shown in Figure 3. A user can refine the suggested companies shown by applying filters and sorting the remaining results. Additionally, a user can select to see the full details for a suggested company, or add it to their list. Filtering and ordering which is applied is stored such that it can be utilized throughout the system. This interaction is one of many examples employed for capturing user intent.

A user can also view a visualization of the list of suggested companies (Figure 4). In this view, the set of suggested companies is displayed on a heat map, showing how similar each suggested company is to companies on the original list. The original companies are listed on the left and suggestions are across the top; each square is colored based on the pairwise similarity of the companies whose intersection it represents. Hovering over a company displays a brief description of it, and users can add companies to their list directly from the visualization.

[1] https://www.crunchbase.com/
[2] https://www.capitaliq.com

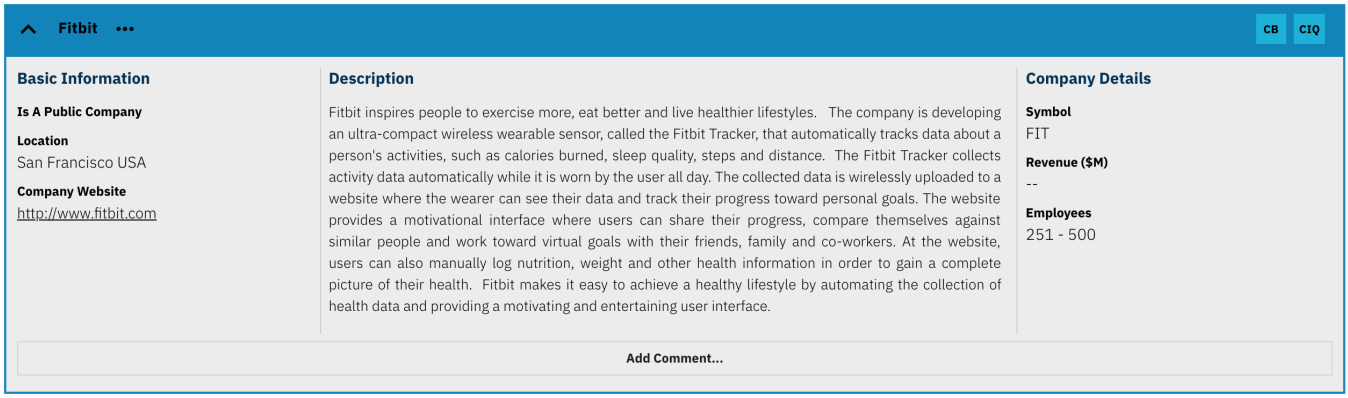

Figure 2: Dropdown showing the summary view of a company.

Figure 3: Set of companies similar in description to a list as a whole. This set can be filtered or sorted by several of the company's properties

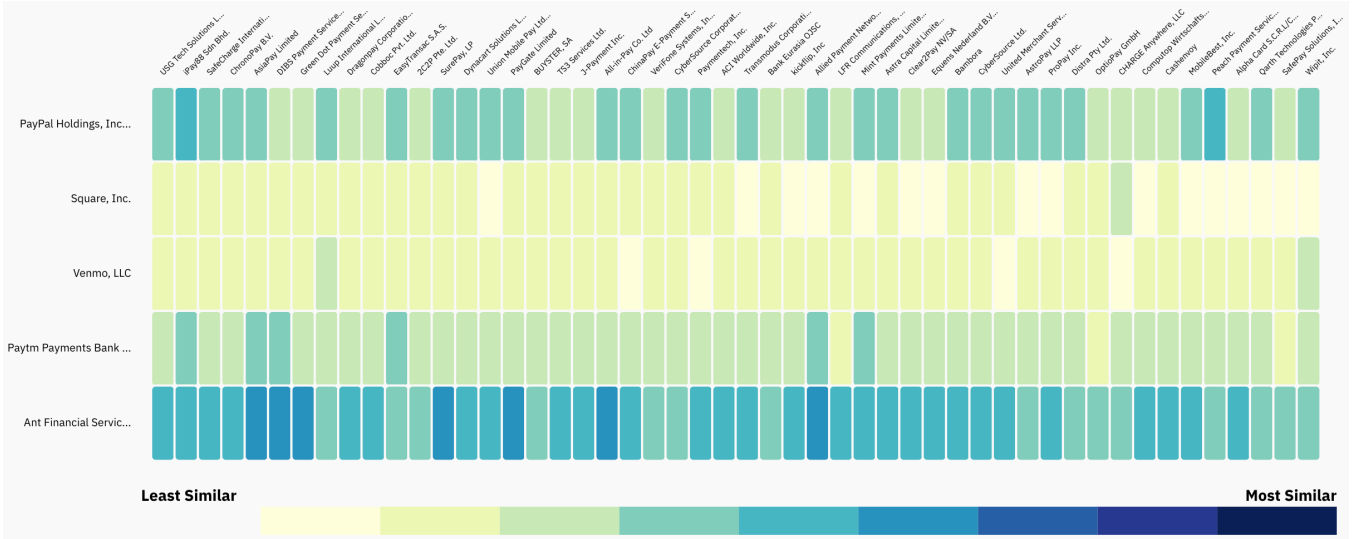

Figure 4: Heatmap view showing similarity of list recommendations to items already on the list

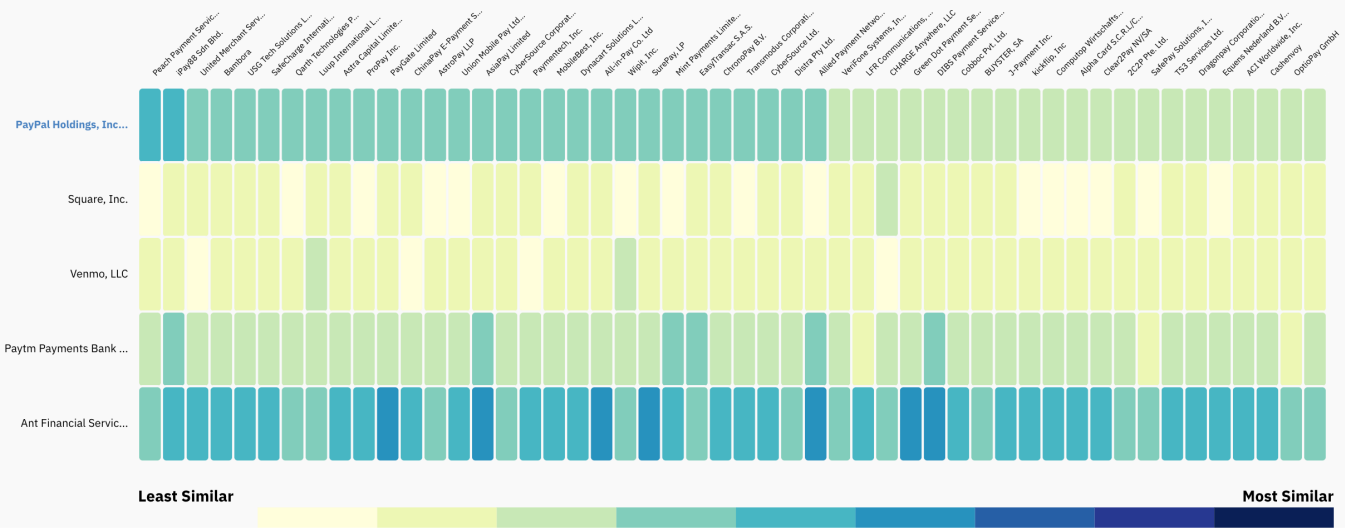

Figure 5: Heatmap view with suggested companies sorted by similarity to the first company already on the list

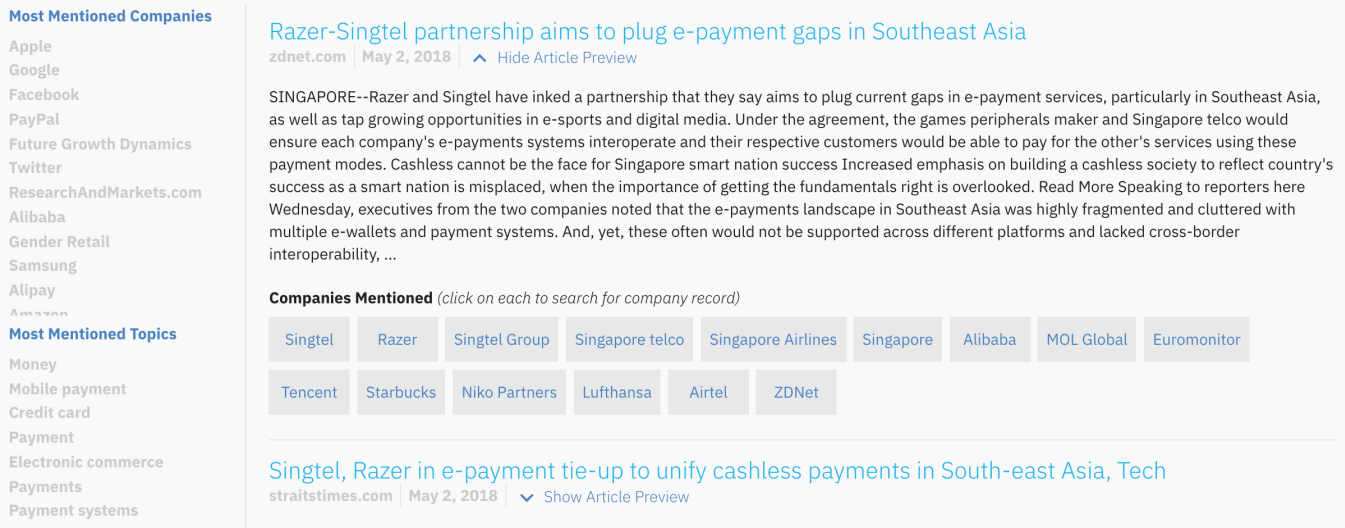

Figure 6: View showing news articles related to key topics associated with the list, with one article entry expanded, showing main text and company entities found in the article

Users can sort the heat map by clicking on any of the list companies on the left, letting them easily see which of the suggested companies are most similar to it (Figure 5).

4.1 News Monitoring

A user can view news stories [1] topically related to the companies on the list as in Figure 6. These topics are selected from a list of thirty automatically extracted concepts; users can also enter concepts manually. Topics which are selected are stored, and can be used for determining user intent.

Users can optionally specify that news results should focus on specific categories of events, such as those related to legal or financial issues. Additionally, we extract and display names of any companies mentioned in the article so they can be easily added to the list.

4.2 Computing Company Similarity

Our company similarity calculations are based on the assumption that similar companies have overlapping key words in their descriptions. We estimate the similarity between two companies via the following steps.

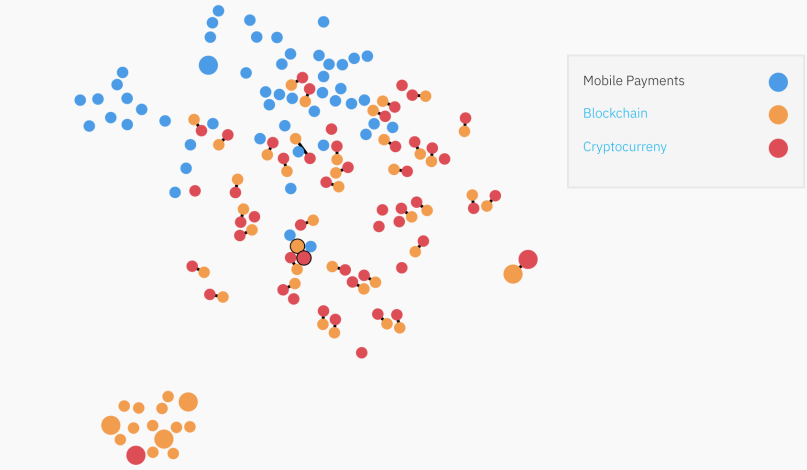

Figure 7: t-SNE visualization showing companies and recommendations for three lists. Those similar to multiple lists are repeated, connected with black lines. Hovering on a node highlights the lists in the legend (right) for which it is suggested.

- Each description is preprocessed to filter out unimportant text, stop words, and boilerplate phrases. We also remove names of specific companies and locations, which generally don't contribute to functional similarity. We identify these words and phrases with a publicly available Natural Language Understanding service[3]. The remaining text is then lemmatized and this process is repeated for the description of all companies in the database.
- A term frequency-inverse document frequency (tf-idf) model is built using the preprocessed descriptions. Each company is now represented by a row in the tf-idf matrix. A tf-idf model has various parameters such as the minimum number of documents a token must appear in for it to be considered in the vocabulary, min_{df}, and similarly, the maximum number of documents a token must appear in to consider it important, max_{df}. As there is no well-defined heuristic to tune these parameters, we defined a surrogate metric to evaluate the performance of the tf-idf model and used this metric to choose the best parameters for our model. (We discuss this further in the next section).
- We define the similarity between two companies as the cosine of their tf-idf vectors.

$$sim(\text{company}_A, \text{company}_B) = \cos(\text{tf-idf}_A, \text{tf-idf}_B)$$

If we want to find the n most similar companies to company A, we need to find the similarity between company A and all companies in the tf-idf matrix. This can be done with a simple matrix multiplication.

$$sim(\text{company}_A, \text{all companies}) = \cos(\text{tf-idf}_A, \text{tf-idf}_{\text{matrix}})$$

This gives us a column of all similarity scores, from which we pick and return the top n corresponding companies.

[3]https://www.ibm.com/watson/services/natural-language-understanding/

4.3 Refinement using ground truth (expert ratings)

As discussed above, we define an evaluation metric to measure the performance of the tf-idf model. For this, we needed ground truth data. We annotated 3000 pairs of company descriptions by assigning each pair a rank of 1 (strong), 2 or 3 (weak). We considered 5 and 7 point scales, but based on the experience our subject matter domain experts had with annotating the ground truth, we settled on the 3 point scale. The simpler 3 point scale also matched how they think about company similarity. To evaluate the performance of the model, we got the similarity scores from the model (as described in the previous section) for all 3000 pairs and then calculated the Spearman's rank correlation coefficient. We iterated over several values of min_{df} and max_{df}, and picked the values that resulted in the best Spearman score.

4.4 Similarity Algorithms for Lists

Our approach to finding the top companies that match a list L of companies is done in three steps.

- **For each company in L, find its similar companies, each with its similarity score** We employ our similarity algorithm based on company description text, as explained in the previous section. We have developed similarity techniques that also take into account structured data, such as number of employees, revenue and location, but this is outside the scope of the work reported here.
- **For each matching company found in step 1, find its aggregate score, based on the individual similarity scores obtained in step 1** Each possibly matching company m_i now has an array of similarity scores, S_i, one for each company in the input list L. An aggregation function is applied to S_i to produce the aggregate score agg_i for m_i. We have experimented with three aggregation schemes: Borda, Average, and Best. Average and Best are derived directly from the

array or scores, where Best uses the highest score and Average uses the average of all scores. Borda is a voting scheme based on the ranks of companies on the match lists: each company in the original list ranks its matches, and the Borda count of a matched company is the sum of all its ranks, such that lower is better.

- **Sort the matching companies by their aggregate scores, agg_i, and return the top ones.**

The initial list input by the user can be used to tune the weights of the similarity function to realize a weighted cosine similarity function [16].

4.5 Visualizing Similarity Across Lists

To help the user find matching companies of interest based on similarity to one or more lists, we employ a t-SNE based [19] visualization. One variant takes the pairwise similarity matrix, converts it to a distance matrix, and uses t-SNE to produce 2D coordinates, so that companies are positioned closest to the ones most similar. When several lists are provided as input (as shown in Figure 7), the visualization helps identify matching companies that are similar to more than one list.

A second variant uses a force-based layout for a graph, in which each company is a node, and two nodes are connected if their similarity is above a given threshold. As we move a threshold slider from 0 to 1, the graph gets sparser, and one can identify neighborhoods and clusters of interest.(as shown in Figure 8) Our implementation uses a back-end service to compute the similarity matrix between all pairs of companies before rendering the graph. By running a simple analytic on the similarity matrix we can derive a reasonable starting value for the slider. For example, we can target an initial density of 30% by choosing a threshold that is just above 30% of the set of values in the matrix.

5 USER FEEDBACK

Initial feedback from business users has been quite positive, described as naturally fitting how they often approach their task and a faster way of identifying companies than the searches and document perusal they typically do. Users often start with several known companies in mind; in fact, in some cases their goal is even described as "finding more companies like X and Y". Users reported that the system recommended companies they knew about and expected to see, as well as making them aware of relevant companies that they had missed in their earlier searches. Some users also said that they would use our system's similar companies feature as a final check for any list they created, to catch companies they may have missed in their other research.

We conducted a preliminary test with 6 subjects creating 3 lists each. Subjects were asked to create lists of 10 companies starting with 3 or 4 seed companies. Subjects were asked to create lists of

(1) Companies producing meatless meat substitutes,
(2) Companies creating smart headphones with AI assistance, and
(3) Companies developing helmets with integrated heads-up displays.

Despite the fact that the corpus of company information for similarity comparisons available to us was limited (both in terms of companies covered and depth of description), all subjects discovered relevant candidate companies that had not been found by subjects performing traditional web searches. This supports the potential use of our tool as an accompaniment to other searches.

6 FUTURE WORK

Our planned future work falls into four general categories: improving text-similarity scoring, developing visualization and interaction mechanisms that help users define and refine their similarity criteria, conducting more formal and extensive user studies to assess and improve the overall usefulness of our generalized similarity service, and demonstrating generality by applying the service to multiple domains.

Thus far, our text similarity scoring has been based mainly on simple variations of tf-idf. While not reported here, preliminary experiments with neural nets suggest that they hold promise. Given recent successes in applying deep learning networks with long short-term memory (LSTM) to natural language processing [35], we plan to pursue such approaches, possibly combining them with other established techniques such as locality-sensitive hashing and other approximate nearest-neighbor algorithms. Distance metrics such as Weighted Cosine Similarity[16] are also something we plan to experiment with.

Another important area for further research and development will be to capture users' mental models through multi-modal, conversational interfaces and to translate those models through the semantic abstraction layer. We plan to add mechanisms for users to see and explicitly refine the system's understanding of their similarity intent including the use of structured information (number of employees, revenue, location etc.) This will include specifying goals for judging similarity and recommending similar items that differ from those based on items already on the list (for example, to indicate the desire to add items from a geography not currently included). Other planned features include learning from negative feedback, for example by adjusting the weights of outlier companies similar to rejected companies, as well as mechanisms for collaborative list building. We will draw on the rich body of literature in relevance feedback for improving the retrieval performance, especially techniques that yield high recall [30]. Additional research will revolve around learning from implicit feedback and interaction traces stored during the search and exploration process in order to improve suggestions.

Furthermore, we plan to conduct additional studies evaluating the quality of lists created using our system by comparison to lists created through other means, including expert ratings of appropriateness of included items, identification of any obvious missed items, and comprehensiveness of the list overall.

Finally, as we believe our approach generalizes readily to applications that require creating lists from among a large set of candid ates, we intend to explore creating lists in other domains such as of people, projects, grant offerings, and studies.

ACKNOWLEDGMENTS

The authors would like to thank Kevin Winpisinger, Dongbo Lin, David Emerson, Ryan Lett, Michael Tannenblatt and Julie MacNaught for their great work on the company similarity solution.

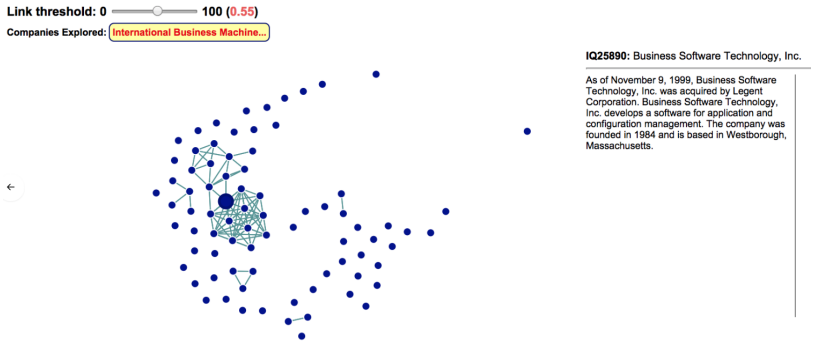

Figure 8: Force directed graph showing similar companies to a single company. The number of companies connected by black lines is controlled by the slider in the upper left. Hovering on a node displays the company descriptive text.

REFERENCES

[1] 2018. Watson Discovery News Service. https://console.bluemix.net/docs/services/discovery/watson-discovery-news.html#watson-discovery-news

[2] Alina Beygelzimer, Sham Kakade, and John Langford. 2006. Cover trees for nearest neighbor. In *Proceedings of the 23rd international conference on Machine learning*. ACM, 97–104.

[3] Antoine Bordes and Jason Weston. 2016. Learning End-to-End Goal-Oriented Dialog. *CoRR* abs/1605.07683 (2016). arXiv:1605.07683 http://arxiv.org/abs/1605.07683

[4] Andrei Z Broder. 1997. On the resemblance and containment of documents. In *Compression and Complexity of Sequences 1997. Proceedings*. IEEE, 21–29.

[5] Marco Cavallo and Çagatay Demiralp. 2018. A Visual Interaction Framework for Dimensionality Reduction Based Data Exploration. *ACM Human Factors in Computing Systems (CHI)* (2018).

[6] Moses S Charikar. 2002. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*. ACM, 380–388.

[7] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[8] Peter Christen. 2012. *Data matching: concepts and techniques for record linkage, entity resolution, and duplicate detection.* Springer Science & Business Media.

[9] Dedre Gentner. 1989. analogical learning. *Similarity and analogical reasoning* (1989), 199.

[10] Dedre Gentner and Arthur B Markman. 1997. Structure mapping in analogy and similarity. *American psychologist* 52, 1 (1997), 45.

[11] Robert L Goldstone. 1994. The role of similarity in categorization: Providing a groundwork. *Cognition* 52, 2 (1994), 125–157.

[12] Daniel Guo, Gokhan Tur, Wen-tau Yih, and Geoffrey Zweig. 2014. Joint semantic utterance classification and slot filling with recursive neural networks. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*. IEEE, 554–559.

[13] Sha Hu, Zhicheng Dou, Xiaojie Wang, Tetsuya Sakai, and Ji-Rong Wen. 2015. Search result diversification based on hierarchical intents. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 63–72.

[14] Herve Jegou, Matthijs Douze, and Cordelia Schmid. 2011. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence* 33, 1 (2011), 117–128.

[15] Clare-Marie Karat, John Vergo, and David Nahamoo. 2002. Conversational interface technologies. In *The human-computer interaction handbook*. L. Erlbaum Associates Inc., 169–186.

[16] Baoli Li and Liping Han. 2013. Distance weighted cosine similarity measure for text classification. In *International Conference on Intelligent Data Engineering and Automated Learning*. Springer, 611–618.

[17] Bing Liu and Ian Lane. 2015. Recurrent neural network structured output prediction for spoken language understanding. In *Proc. NIPS Workshop on Machine Learning for Spoken Language Understanding and Interactions*.

[18] Bing Liu and Ian Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. *arXiv preprint arXiv:1609.01454* (2016).

[19] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[20] Andrew McCallum, Dayne Freitag, and Fernando CN Pereira. 2000. Maximum Entropy Markov Models for Information Extraction and Segmentation.. In *Icml*, Vol. 17. 591–598.

[21] Martino Mensio, Giuseppe Rizzo, and Maurizio Morisio. 2018. Multi-turn QA: A RNN Contextual Approach to Intent Classification for Goal-oriented Systems. In *Companion of the The Web Conference 2018 on The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 1075–1080.

[22] Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Hakkani-Tur, Xiaodong He, Larry Heck, Gokhan Tur, Dong Yu, et al. 2015. Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23, 3 (2015), 530–539.

[23] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[24] Sharon Oviatt, Phil Cohen, Lizhong Wu, Lisbeth Duncan, Bernhard Suhm, Josh Bers, Thomas Holzman, Terry Winograd, James Landay, Jim Larson, et al. 2000. Designing the user interface for multimodal speech and pen-based gesture applications: state-of-the-art systems and future research directions. *Human-computer interaction* 15, 4 (2000), 263–322.

[25] Christian Raymond and Giuseppe Riccardi. 2007. Generative and discriminative algorithms for spoken language understanding. In *Eighth Annual Conference of the International Speech Communication Association*.

[26] Edwina L Rissland. 2006. Ai and similarity. *IEEE Intelligent Systems* 21, 3 (2006), 39–49.

[27] Gerard Salton, Anita Wong, and Chung-Shu Yang. 1975. A vector space model for automatic indexing. *Commun. ACM* 18, 11 (1975), 613–620.

[28] Ruhi Sarikaya, Geoffrey E Hinton, and Bhuvana Ramabhadran. 2011. Deep belief nets for natural language call-routing. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 5680–5683.

[29] Christopher-J Schild and Simone Schultz. 2016. Linking Deutsche Bundesbank Company Data using Machine-Learning-Based Classification. In *Proceedings of the Second International Workshop on Data Science for Macro-Modeling*. ACM, 10.

[30] Justin JongSu Song and Wookey Lee. 2017. Relevance maximization for high-recall retrieval problem: finding all needles in a haystack. *The Journal of Supercomputing* (2017), 1–24.

[31] Dinoj Surendran and Gina-Anne Levow. 2006. Dialog act tagging with support vector machines and hidden Markov models. In *Ninth International Conference on Spoken Language Processing*.

[32] Jingdong Wang, Heng Tao Shen, Jingkuan Song, and Jianqiu Ji. 2014. Hashing for similarity search: A survey. *arXiv preprint arXiv:1408.2927* (2014).

[33] Yair Weiss, Antonio Torralba, and Rob Fergus. 2009. Spectral Hashing. In *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou (Eds.). Curran Associates, Inc., 1753–1760. http://papers.nips.cc/paper/3383-spectral-hashing.pdf

[34] Kaisheng Yao, Baolin Peng, Yu Zhang, Dong Yu, Geoffrey Zweig, and Yangyang Shi. 2014. Spoken language understanding using long short-term memory neural networks. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*. IEEE, 189–194.

[35] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. 2017. Comparative study of cnn and rnn for natural language processing. *arXiv preprint arXiv:1702.01923* (2017).