# Corpus Compression for Deep Reinforcement Learning in Natural Language Environments

Zhiwen Tang
Infosense, Department of Computer Science
Georgetown University
zt79@georgetown.edu

Grace Hui Yang
Infosense, Department of Computer Science
Georgetown University
huiyang@cs.georgetown.edu

## ABSTRACT

Deep reinforcement learning (DRL) offers natural solutions to interactive artificial intelligence (AI) agents, for DRL's ability of adaptation and exploration. Many existing methods use a search engine's retrieval function as their RL agent's action to retrieve information and explore in an information space. What is neglected is that off-the-shelf retrieval functions are optimized over precision at top ranks and this would prevent the agent from knowing the global picture and visiting all areas in the state space. In this paper, we propose to compress an entire text corpus into a global low-dimensional representation, which grants an agent with global access to the full state space. We experiment on the Text REtrieval Conference (TREC) Dynamic Domain (DD) Track and the results show that our method outperforms the state-of-the-art dynamic search (DS) systems. A complete version of this paper has been published in the Deep Reinforcement Learning Workshop at NeurIPS 2019.

**ACM Reference Format:**
Zhiwen Tang and Grace Hui Yang. 2020. Corpus Compression for Deep Reinforcement Learning in Natural Language Environments. In *The 1st Workshop on Deep Reinforcement Learning for Information Retrieval (DRL4IR '20), July 30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION

Research in interactive artificial intelligence (AI) agents aims to make the agents assist a human user to accomplish a goal-oriented task [7, 12, 20]. Deep reinforcement learning has been applied to interactive AI agents. Prior work on this topic has investigated the use of bandits-based [7], value-based [17], and policy-based [9] RL methods. Many interactive agents are retrieval-based systems. It means, they often use a search engine to retrieve documents and this forms the RL agents' actions to explore in an information space. The retrieval-based interactive systems include multi-turn Question Answering (QA) [12], dialogue systems [3], and dynamic search systems [20].

These systems share similar designs. The environments often consist of a repository of documents (or knowledge) and a human user. Also, the agents' actions often include two steps. The first is to reformulate a new query or ask a new question, based on user responses. The second is to retrieve relevant information to those queries via some off-the-shelf retrieval tools.

Such a pipeline is a convenient use of existing search engine techniques. However, most existing retrieval functions are optimized over precision at top ranks, which is determined by a human's limited cognitive load when examining the results. This bias is engraved in almost all ready-to-use retrieval tools. The effect is that results that are good but not as optimal would be difficult to show up. It is thus difficult for the agent to be aware of the global picture of the state space, for the search is so restricted by the top results. This is different from how an RL agent is treated in AI, where the agent is always aware of the global status, e.g. AlphaGo knows the game board. The inadequate knowledge of the global picture makes the learning of the RL agent quite difficult because the agent could not "explicitly consider the *whole* problem of a goal-oriented" process [16].

In this paper, we propose to create a global representation to encode an entire natural language corpus, enabling the RL agent to gain access to full state space. Using dynamic search (DS) as an illustrating example, we present a deep reinforcement learning framework that can be used in natural language environments for interactive AI agents.

One option to produce natural language (NL) global representation is Doc2vec [6]. It has been used to construct corpus-level representation in deep learning. Doc2vec transforms a high-dimensional discrete representation of documents into low-dimensional continuous vectors by predicting the central word with its neighboring words and the corresponding document vector. Unfortunately, however, doc2vec could not be able to solve a problem known as *crowding* [1]. Crowding refers to the situation where multiple high-dimensional data points are collapsed into one after dimension reduction. It suggests that if two data points belong to two different classes, after collapsing it would be impossible to tell them apart. In our case where each data point represents either a relevant or irrelevant document, this would be an issue.

Therefore, in this paper, we propose a different solution to retrieval-based interactive agents. In our corpus-level end-to-end exploration algorithm (CE3), at each time step, a text corpus is compressed into a single global representation and used to support full exploration in the entire state space.

Experiments on the Text REtrieval Conference (TREC) Dynamic Domain 2017 Track demonstrate that our method significantly outperforms previous DS approaches. It is also shown that our method is able to quickly adjust search trajectories and recover from losses in early interactions. Given the fundamental issues

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
*DRL4IR '20, July 30, 2020, Virtual Event, China*
© 2020 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

that it addresses, we believe CE3's success can be extended to other interactive AI systems accessing information with retrieval functions.

## 2 RELATED WORK

Knowing a global model that oversees an entire document collection has shown substantial benefits in traditional IR. For instance, corpus-level clusters was developed via K-means in [8] or Latent Dirichlet Allocation (LDA) [19] to improve retrieval scores. In this paper, we encode the content of the entire corpus and the user's search history into a global state representation.

KB-InfoBot [3] is a dialogue-based system to find movies from a large movie knowledge base (KB). Similar to us, KB-InfoBot used global representation for *all* movies to represent the states. To do so, it estimated the target movie by modeling a global distribution over the entire set of movie entities. The difference is that their work is for data stored in a structural database and our work is for largely available natural language free texts.

Many great breakthroughs in Natural Language Processing (NLP) are built upon word2vec [11]. Its derivation doc2vec [6] transforms high-dimensional discrete representation of documents into low-dimensional continuous vectors. It maximizes the following log probability: $\frac{1}{KT}\sum_{k=1}^{K}\sum_{t=m}^{T-m}\log p(w_{k,t}|w_{k,t-m},...,w_{k,t+m},v_k)$ where $v_k$ is the document vector of the $k^{th}$ document and $w_{k,t}$ is the vector of the $t^{th}$ word in the $k^{th}$ document. The document vector $v_k$ is a compressed low dimensional representation that seems desirable. It is initialized randomly and optimized via self-supervision. However, the optimization goal does not put constraints on its post-optimization distribution. Thus, doc2vec would not be able to solve the crowding problem [1]. In our experiments, doc2vec performs poorly. Instead, we choose to use the t-Distributed Stochastic Neighbor Embedding (t-SNE) method [10].

## 3 THE APPROACH

In this paper, we propose to compress an entire text corpus into a global low-dimensional representation and keep it around all the time. In this way, we can then enable an RL agent to gain accesses to the full state space. It is essential for an RL agent because not being able to reach documents in under-explored areas would mean not being able to recover from early bad decisions. We summarize our procedure of creating the global representation into three steps.

First, each document is split into topic-coherent segments. The latest advances in Neural Information Retrieval (NeuIR) have demonstrated the effectiveness of exploiting topic structures [18]. In this work, we follow [18] for segmenting and standardizing documents. Each document is first segmented into topical coherent blocks then standardized to has a fixed $B$ number of segments.

Second, each segment is compressed into a much lower dimension $n$ ($n \ll W$). Bag-of-Words (BoW) is used as feature vector for a segment and is of the same size as the vocabulary ($W$). One challenge is that after the compression the documents would be crowded and it would be difficult to tell apart the relevant documents from the irrelevant ones. To address this issue, We employ t-SNE [10] for dimension reduction.

Assume the high-dimensional input $\boldsymbol{x}_* \in \mathbb{R}^W$ follows the Gaussian distribution and there are two random input data points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. The probability that $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are neighboring to each other is $p(\boldsymbol{x}_i,\boldsymbol{x}_j) = \frac{exp(-||\boldsymbol{x}_i-\boldsymbol{x}_j||^2/2\sigma^2)}{\sum_{k\neq l} exp(||\boldsymbol{x}_k-\boldsymbol{x}_l||^2/2\sigma^2)}$. The algorithm then maps those data points in the high dimensional space $\boldsymbol{x}_*$ to $\boldsymbol{y}_*$, which is in a space with much lower dimensionality, $\mathbb{R}^n$. Suppose $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are projected into the lower dimension as $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$. The probability that $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ are still neighboring to each other is then $q(\boldsymbol{y}_i,\boldsymbol{y}_j) = \frac{(1+||\boldsymbol{y}_i-\boldsymbol{y}_j||^2)^{-1}}{\sum_{k\neq l}(1+||\boldsymbol{y}_k-\boldsymbol{y}_l||^2)^{-1}}$. KL divergence measures the difference between two distributions, $p(*)$ and $q(*)$. Minimizing it yields the solution

$$L_{tsne}(\boldsymbol{y}||\boldsymbol{x}) = \sum_i \sum_j p(\boldsymbol{x}_i,\boldsymbol{x}_j)\log\frac{p(\boldsymbol{x}_i,\boldsymbol{x}_j)}{q(\boldsymbol{y}_i,\boldsymbol{y}_j)} \quad (1)$$

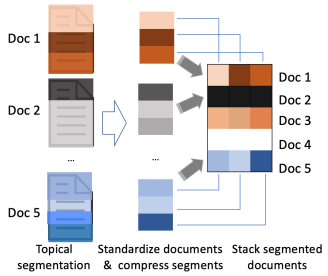


**Figure 1: Global representation of a toy corpus (of 5 documents): Documents are segmented and standardized following [18]. Similar colors suggest similar contents. Document 2 is darkened after being visited. Document 4 is currently selected by the RL agent and highlighted with white.**

Third, all the documents are stacked together to form a global representation. It is denoted by $C$ and its dimensions are $C \times B \times n$. Here $C$ is the number of documents, $B$ is the number of segments per document, and $n$ is the reduced feature dimension.

Fig. 1 illustrates the global representation of a toy corpus. In this global representation $C$, each row represents a document and each column represents a segment in this document. For generality, we make no assumption about the stacking order of documents. It is because the RL agent is expected to complete the search task even when it is dealing with randomly ordered documents.

The states are constructed using this global representation. The state at time $t$, $\boldsymbol{s}_t$, consists of two parts: i) this corpus-level representation $C$ and ii) the retrieval history of documents from time 1 to $t-1$:

$$\boldsymbol{s}_t = S(C, \mathcal{D}_1 \cup \mathcal{D}_2 \cup ...\mathcal{D}_i... \cup \mathcal{D}_{t-1}) \quad (2)$$

where $\mathcal{D}_i$ is the set of documents retrieved at time $i$.

The action of the RL agent is the parameter of a differentiable ranking function, which estimates the relevance of each document at each time step:

$$score_{i,t} = f(\boldsymbol{a}_t^{\boldsymbol{\theta}}, d_i) = \sum_{j=1}^{B} \boldsymbol{y}_{ij} \cdot \boldsymbol{a}_t^{\boldsymbol{\theta}} \quad (3)$$

where $\boldsymbol{a}_t^{\boldsymbol{\theta}}$ is the sampled action and $\boldsymbol{y}_{ij}$ is the feature vector of the $j^{th}$ segment in $d_i$ after compression.

Our RL framework uses a state-of-the-art policy-gradient method, Proximal Policy Optimization (PPO) [15]. The RL agent consists of two networks, a value network and a policy network. PPO optimizes

Initialize $\boldsymbol{\theta}$;
**for** *iteration = 1, 2, ...* **do**
  **for** *t=1,2, ... , T* **do**
    read in the global representation $\boldsymbol{s}_t$;
    sample action $\boldsymbol{a}_t^{\boldsymbol{\theta}} = \pi(\boldsymbol{s}_t, \boldsymbol{\theta}_t)$;
    **for** *i=1,2, ... , C* **do**
      estimate a relevance score for document $d_i$:
      $score_{i,t} = f(\boldsymbol{a}_t^{\boldsymbol{\theta}}, d_i)$ (Eq. 3)
    **end**
    rank the documents by $score_{*,t}$ and return the top-ranked documents $\mathcal{D}_t$;
    compute $r_t = \sum_{d_i \in \mathcal{D}_t \setminus (\mathcal{D}_1 \cup \mathcal{D}_2 \cup ... \cup \mathcal{D}_{t-1})} rel(d_i)$;
    mark returned documents as visited, generate next state $\boldsymbol{s}_{t+1}$;
    Compute advantage $\hat{A}_t$;
  **end**
  Optimize $L(\boldsymbol{\theta})$ (Eq. 4) w.r.t. $\boldsymbol{\theta}$;
**end**

**Algorithm 1:** CE3.

| Topic (DD17-10) | Leaning Towers of Pisa Repairs |
|---|---|
| Subtopic 1 (id: 321) | Tourism impact of repairs/closing |
| Subtopic 2 (id: 319) | Repairs and plans |
| Subtopic 3 (id: 320) | Goals for future of the tower |
| Subtopic 4 (id: 318) | Closing of tower |

**Table 1: Example Search Topic.**

the following objective function with sampled trajectories.

$$L(\boldsymbol{\theta}) = \hat{\mathbb{E}}_t[L_t^{Policy}(\boldsymbol{\theta}) - c_1 L_t^{Value}(\boldsymbol{\theta})] \tag{4}$$

where $c_1$ is coefficients. $L_t^{Value}(\boldsymbol{\theta})$ is the mean squared error between the estimated state value and the targeted state value. $L^{Policy}$ is a pessimistic bound of the effectiveness of policy network. It is defined as $L_t^{Policy}(\boldsymbol{\theta}) = \hat{\mathbb{E}}_t[\min(\rho_t(\boldsymbol{\theta})\hat{A}_t, clip(\rho_t(\boldsymbol{\theta}), 1-\epsilon, 1+\epsilon)\hat{A}_t)]$.

Algorithm 1 describes the CE3 algorithm. It starts by sampling actions by the RL agent. The documents are then ranked by their estimated relevance scores calculated based on the action vectors. The top-ranked ones are shown to the user. The user then examines the documents and submits feedback, which is used to derive the immediate reward $r_t$. The algorithm employs stochastic gradient ascent to optimize both the policy network and the value network. The process continues until no better policy could be found.

## 4 EXPERIMENTAL SETTINGS

Our experiments are based on the TREC 2017 Dynamic Domain Track. The TREC 2017 DD Track has the most complete set of ground truth answers as compared to 2015 and 2016. The TREC 2017 DD Track used LDC New York Times collections [14] as its corpus. The Track released 60 search tasks created by human assessors. Each task consists of multiple hierarchically-organized subtopics. The subtopics were not made available to the participating DS systems. An example DD search topic DD17-10 is shown in Table 1. A simulated user[1] issues a starting query, and then provides feedback for all the subsequent runs of retrievals. The feedback includes graded relevance judgments in the scale of 0 to 4, and points out relevant passages for the documents. An example DD interaction between the system and the user is shown in Table 2.

[1]https://github.com/trec-dd/trec-dd-jig

| | Search DD17-10 |
|---|---|
| User: | *Leaning Towers of Pisa Repairs* |
| System: | Return document 0290537 |
| User: | Non-relevant document. |
| System: | Return document 0298897 |
| User: | Relevant on subtopic 320 with a rating of 2, *"No one doubts that it will collapse one day unless preventive measures are taken."* |
| System: | Return document 0984009 |
| User: | Relevant on subtopic 318 with a rating of 4, *"The 12th-century tower was closed to tourists in 1990 for fear it might topple."* |

**Table 2: Example Interaction History.**

### 4.1 Metrics and Baselines

The evaluation focuses on gaining relevant information throughout the whole process. We adopt multiple metrics to evaluate the approaches from various perspectives. **Aspect recall** [5] measures subtopic coverage. **Precision** and **Recall** are ratios of correctly retrieved documents over the retrieved document set or the entire correct set. **Normalized Session Discounted Cumulative Gain (nsDCG)** evaluates the graded relevance for a ranked document list, with heavier weights put on the early retrieved ones [4].

We compare CE3 with its variant **CE3 (doc2vec)**, which uses doc2vec to construct the compressed representation of each segment. Apart from that, we also compare with the most recent dynamic search systems from TREC DD 2017 submissions. We pick the top submitted run from each team: **Galago** [2], which repeatedly retrieves document using the same query without using any feedback; **RF**, a relevance feedback algorithm used by [13]; **DIV**, a search result diversification algorithm used by [21]; and **DQN**, which selects query reformulation actions at each time step [17].

### 4.2 Parameters

We construct a collection for each search topic by mixing relevant documents and irrelevant documents at a ratio of 1:1. The corpus size $C$ ranges from tens to thousands. The following configuration yields the best performance: In the global representation $C$, $n$ is set to 3, $B$ is set to 20. Coefficients $c_1$ in Eq. 4 is 0.5. Both policy network and value network are composed of two layers of CNNs and one MLP. The first CNN consists of eight $2 \times 2$ kernels; while the second consists of sixteen. The hidden layer of MLP consists of 32 units.

## 5 RESULTS ANALYSIS

**CE3 v.s. CE3 (doc2vec):** From Fig. 2, it comes to our attention that CE3 and CE3 (doc2vec) show similar trends in all metrics, but the latter is left far behind by CE3. We then investigate if they are capable of exploring different documents over time. We discover that CE3 retrieves much less duplicate documents than CE3 (doc2vec) does. Table 3 reports $\frac{|\mathcal{D}_t \cap (\mathcal{D}_1 \cup \mathcal{D}_2 \cup ... \cup \mathcal{D}_{t-1})|}{|\mathcal{D}_t|}$, the percentage of duplicate

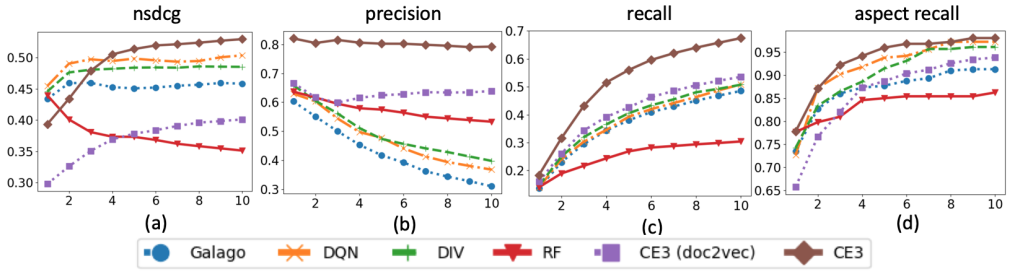


**Figure 2: Experiment results in the first 10 search iterations.**

| Time step | t=1 | t=2 | t=3 | t= 4 | t=5 | t=6 | t=7 | t=8 | t=9 | t=10 |
|---|---|---|---|---|---|---|---|---|---|---|
| CE3 (doc2vec) | 0.0% | 11.7% | 18.0% | 30.0% | 35.0% | 34.3% | 30.0% | 25.0% | 25.7% | 25.0% |
| CE3 | 0.0% | 3.0% | 1.7% | 1.0% | 3.0% | 0.0% | 1.0% | 0.3% | 3.0% | 0.0% |

**Table 3: Percentage of duplicate documents.**

documents being retrieved, for the two CE3 variants. We believe it is due to how they compress the feature vectors in a segment. Doc2vec makes no assumption about the data distribution after compression. Vectors trained by doc2vec are probably crowded together and yield more duplicate results. On the contrary, t-SNE helps CE3 separate relevant documents from the irrelevant ones, which contributes to the success of CE3.

**CE3 v.s. other baselines:** Fig. 2 also shows that CE3 outperforms all others in recall (Fig. 2c) and aspect recall (Fig. 2d) at all time. It suggests that our RL agent is able to explore more areas in the action space than the rest. CE3 also performs quite impressive in precision (Fig. 2b). As a search episode develops, all other approaches show declined performance, except CE3 stays strong at all iterations, which indicates that global representation makes the exploration more effective. Moreover, results on nsDCG (Fig. 2a) reveal that, although CE3 does not score as high as other methods at beginning, CE3 largely outperforms the rest at the end of episode. Those well-tuned ranking-sensitive retrieval functions seems not to be able to adapt well when the number of interactions increases.

## 6 CONCLUSION

Using Dynamic Search (DS) as an example, this paper shows how to represent an entire information space formed by free text documents to support deep reinforcement learning in natural language environments. We propose to maintain a global representation of entire corpus at all time. Corpus-level compression is achieved by using dimension reduction via t-SNE. The experimental results have demonstrated our method's strong performance superior to other state-of-the-art DS systems.

## ACKNOWLEDGMENTS

This research was supported by U.S. National Science Foundation IIS-145374. Any opinions, findings, conclusions, or recommendations expressed in this paper are of the authors, and do not necessarily reflect those of the sponsor.

## REFERENCES

[1] James Cook, Ilya Sutskever, Andriy Mnih, and Geoffrey E. Hinton. 2007. Visualizing Similarity Data with a Mixture of Maps. In *AISTATS '07*.
[2] W. Bruce Croft, Donald Metzler, and Trevor Strohman. 2009. *Search Engines - Information Retrieval in Practice*. Pearson Education.
[3] Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2017. Towards End-to-End Reinforcement Learning of Dialogue Agents for Information Access. In *ACL '17*.
[4] Kalervo Järvelin, Susan L. Price, Lois M. L. Delcambre, and Marianne Lykke Nielsen. 2008. Discounted Cumulated Gain Based Evaluation of Multiple-Query IR Sessions. In *ECIR '08*.
[5] Eric Lagergren and Paul Over. 1998. Comparing Interactive Information Retrieval systems Across Sites: The TREC-6 Interactive Track Matrix Experiment. In *SIGIR '98*.
[6] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *ICML '14*.
[7] Cheng Li, Paul Resnick, and Qiaozhu Mei. 2016. Multiple Queries as Bandit Arms. In *CIKM '16*.
[8] Xiaoyong Liu and W. Bruce Croft. 2004. Cluster-based retrieval using language models. In *SIGIR '04*.
[9] Jiyun Luo, Sicong Zhang, and Hui Yang. 2014. Win-win search: dual-agent stochastic game in session search. In *SIGIR '14*.
[10] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR* 9, Nov (2008), 2579–2605.
[11] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS '13*.
[12] Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A Conversational Question Answering Challenge. *TACL* 7 (2019), 249–266.
[13] Kristine Rogers and Douglas W. Oard. 2017. UMD_CLIP: Using Relevance Feedback to Find Diverse Documents for TREC Dynamic Domain 2017. In *TREC '17*.
[14] Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia* 6, 12 (2008), e26752.
[15] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).
[16] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction* (second ed.). The MIT Press.
[17] Zhiwen Tang and Grace Hui Yang. 2017. A Reinforcement Learning Approach for Dynamic Search. In *TREC '17*.
[18] Zhiwen Tang and Grace Hui Yang. 2019. DeepTileBars: Visualizing Term Distribution for Neural Information Retrieval. In *AAAI '19*.
[19] Xing Wei and W. Bruce Croft. 2006. LDA-based document models for ad-hoc retrieval. In *SIGIR '06*.
[20] Grace Hui Yang, Zhiwen Tang, and Ian Soboroff. 2017. TREC 2017 Dynamic Domain Track Overview. In *TREC '17*.
[21] Weimin Zhang, Yaokang Hu, Rongqian Jia, Xianfa Wang, Le Zhang, Yue Feng, Sihao Yu, Yuanhai Xue, Xiaoming Yu, Yue Liu, and Xueqi Cheng. 2017. ICTNET at TREC 2017 Dynamic Domain Track. In *TREC '17*.