ISSN 2383-630X(Print) / ISSN 2383-6296(Online)
Journal of KIISE, Vol. 49, No. 1, pp. 78-88, 2022. 1
https://doi.org/10.5626/JOK.2022.49.1.78

격자 탐색을 통한 확장 학습 블룸 필터의 거짓 양성 비율 개선

(Improving False Positive Rate of Extended Learned Bloom Filters Using Grid Search)

양 수 현 †　　김 형 주 ††
(Soohyun Yang)　　(Hyungjoo Kim)

요 약 블룸 필터는 집합을 표현하는 자료구조로 데이터의 포함 여부에 대해서 반환하는 역할을 수행한다. 단, 공간을 적게 사용하는 대가로 거짓 양성을 반환하는 경우가 존재한다. 학습 블룸 필터는 기존의 블룸 필터에 추가적으로 기계학습 모델을 전처리 과정에 사용하여 거짓 양성 비율을 개선하는 방법이다. 즉, 학습 블룸 필터는 기계학습 모델로 일부의 데이터를 저장하고, 모델이 저장하지 못하는 데이터는 보조 필터에 저장한다. 보조 필터는 블룸 필터를 그대로 사용하는 방법도 존재하지만, 본 논문에서의 보조 필터는 블룸 필터와 학습 해시 함수를 같이 사용하는 학습 블룸 필터에 대해서 살펴보고 이를 확장 학습 블룸 필터라고 부른다. 학습 해시 함수는 전처리 과정에서 사용하던 기계학습 모델의 출력값을 해시 함수로 사용하는 방법이다. 본 논문에서는 격자 탐색을 통해서 확장 학습 블룸 필터의 거짓 양성 비율을 개선하는 방법을 제안한다. 이는 학습 해시 함수의 비율을 나타내는 초매개변수의 값을 늘려나가며 가장 낮은 거짓 양성 비율을 가지는 확장 학습 블룸 필터를 탐색하는 방법이다. 결과적으로, 100,000개 이상의 데이터를 저장해야하는 실험 환경에서는 격자 탐색을 통해서 선택된 확장 학습 블룸 필터가 기존의 학습 블룸 필터보다 20% 개선된 거짓 양성 비율을 가질 수 있음을 실험적으로 보인다. 추가적으로, 학습 해시 함수에 사용되는 인공신경망 모델의 출력값이 32비트 부동소수점인 경우에 거짓 음성 오류 문제가 발생할 수 있음을 보이고, 이를 64비트 부동소수점으로 변경하면 해결됨을 보인다. 마지막으로, 10,000개의 데이터를 질의하는 실험 환경에서 인공신경망 모델의 구조를 조정하여 20KB의 공간을 절약하고 동일한 거짓 양성 비율을 갖는 확장 학습 블룸 필터를 만들 수 있음을 보인다. 단, 20KB의 공간을 절약하는 대가로 질의 시간이 2% 늘어난 것을 실험적으로 보인다.

키워드: 자료구조, 블룸 필터, 학습 블룸 필터, 학습 해시 함수, 학습 인덱스, 기계학습, 인공신경망, 거짓 양성 비율, 격자 탐색, 초매개변수

Abstract Bloom filter is a data structure that represents a set and returns whether data is included or not. However, there are cases in which false positives are returned at the cost of using less space. The learned bloom filter is a variation of the bloom filter, that uses a machine learning model in the pre-processing process to improve the false-positive rate. The learned bloom filter stores some data in the machine learning model, and the leftover data is stored in the auxiliary filter. An

† 비 회 원 : 서울대학교 컴퓨터공학부 학생(Seoul Nat'l Univ.)
shyang@idb.snu.ac.kr
(Corresponding author임)
†† 종신회원 : 서울대학교 컴퓨터공학부 교수
hjk@snu.ac.kr

논문접수 : 2021년 7월 16일
(Received 16 July 2021)
논문수정 : 2021년 11월 15일
(Revised 15 November 2021)
심사완료 : 2021년 12월 15일
(Accepted 15 December 2021)

정보과학회논문지 제49권 제1호(2022. 1)

auxiliary filter can be implemented by using a bloom filter only, but in this paper, we use the bloom filter and the learned hash function, and this is called an extended learned bloom filter. The learned hash function uses the output value of the machine learning model as a hash function. In this paper, we propose a method that improves the false positive rate of the extended learned bloom filter through grid search. This method explores the extended learned bloom filter with the lowest false positive rate, by increasing the hyperparameter that represents the ratio of the learned hash function. As a result, we experimentally show that the extended learned bloom filter selected through grid search, can have a 20% improvement in false-positive rate compared to the learned bloom filter, in the experiment that needs more than 100,000 data to store. In addition, we also show that the false negative error may occur in the learned hash function by the use of 32-bit floating points in the neural network model. This can be solved by changing the floating points to 64-bit. Finally, we show that in an experiment where we query 10,000 data, we can adjust the structure of the neural network model to save 20KB of space and create an extended learned bloom filter with the same false-positive rate. However, the query time is increased by 2% at the cost of saving 20KB of space.

Keywords: data structure, bloom filter, learned bloom filter, learned hash function, learned index, machine learning, neural network, false positive rate, grid search, hyperparameter

1. 서 론

최근에 The Case for learned index structures[1]라는 이름의 논문으로 블룸 필터를 포함한 여러 종류의 인덱스가 기계학습을 사용하면 성능이 개선될 수 있음을 보였다. 해당 논문에서는 기계학습 모델을 해시 함수로 사용하는 방법인 학습 해시 함수에 대한 소개가 있으나, 학습 해시 함수에 대한 추가적인 연구가 없는 상태이다.

본 논문에서는 학습 해시 함수를 사용하는 학습 블룸 필터를 확장 학습 블룸 필터로 부르고, 이에 대한 거짓 양성 비율을 개선하는 방법을 제안한다. 학습 해시 함수의 비율을 나타내는 초매게변수를 생성하고, 해당 값에 격자 탐색을 수행하여 개선된 거짓 양성 비율을 갖는 확장 학습 블룸 필터를 탐색하는 방법이다. 이는 제한된 공간 내에서 학습 해시 함수가 가장 좋은 성능을 갖는 지점을 찾는 방법으로 볼 수 있다.

거짓 양성 비율은 블룸 필터에서 입력되지 않았지만, 입력되었다는 결과를 나타내는 오류의 비율이다. 제한된 공간 안에서 더 작은 거짓 양성 비율을 갖기 위한 연구는 쿠쿠 필터[2], XOR 필터[3]에 의해서 진행되었으며, 거짓 양성 비율의 개선과 관련 연구는 계속해서 진행되고 있는 상태이다.

본 논문에서 제안하는 격자 탐색은 구현이 간단하여, 적용이 쉽다는 장점이 있다. 또한, 학습 해시 함수가 기계학습 모델 그리고 데이터에 종속적인 성능을 갖는 특성에 의해서 격자 탐색은 반드시 필요한 과정이라 할 수 있다.

논문의 구조는 확장 학습 블룸 필터를 이해하기 위한 관련된 연구에 대해서 먼저 설명한다. 이후에는 확장 학습 블룸 필터와 격자 탐색 알고리즘에 대해서 알아본다. 추가적으로, 확장 학습 블룸 필터를 구현하며 발생할 수 있는 거짓 양성 문제에 대해서 다루며, 모델 조정을 통해서도 거짓 양성 비율을 개선할 수 있음을 보인다. 또한, 학습 해시 함수가 어떠한 원리로 거짓 양성 비율을 개선하는지 알아본다. 마지막으로, 확장 학습 블룸 필터와 관련된 실험 결과에 대해서 살펴보는 형식으로 진행된다.

2. 관련 연구

블룸 필터[4]는 1970년 B. H. Bloom에 의해서 제안된 확률적인 자료구조이다. 블룸 필터는 집합을 표현하며, 질의된 원소가 집합 내부에 존재하는지 확인하는 기능을 수행한다. 블룸 필터는 집합 S를 표현하기 위해서 k개의 해시 함수, m 크기의 비트 벡터를 사용한다. k개의 해시 함수는 원소가 저장될 비트 벡터의 인덱스 값을 반환한다. 비트 벡터의 공간은 1 또는 0으로 표현되고, 데이터의 존재 여부를 표현할 때 사용된다. 즉, 블룸 필터는 해시 함수와 비트 벡터를 함께 사용하여 집합 S의 원소들에 대한 존재 여부를 확인하는 기능을 수행한다.

블룸 필터를 생성하기 위해서는 집합 S에 있는 각 원소에 대해서 k개의 해시 함수를 통해서 얻은 비트 벡터의 인덱스 값을 1로 갱신하면 된다. 블룸 필터에서 질의하기 위해서는 원소가 k개의 해시 함수에 대해서 모두 1이라는 비트 벡터의 값을 가지게 되면 해당 원소는 집합에 포함됨을 의미하고, 한번이라도 0이라는 비트 벡터의 값을 가지게 되면 집합에 포함되지 않음을 의미한다. 블룸 필터는 집합에 포함되어 있지 않는 원소임에도 불구하고 가끔씩은 집합에 포함되어 있음을 반환하는 경우가 존재하며, 이를 거짓 양성이라고 한다. 거짓 양성은 제한된 m 크기의 비트 벡터에 표현하고자 하는 집합 S의 크기가 커질수록 빈번하게 발생한다.

다음은 블룸 필터에 대한 예시를 살펴보도록 한다. 그림 1에서는 집합 S를 표현하기 위해서 1개의 해시 함수를 사용하고, 크기 7의 비트 벡터를 사용하고 있다. 블룸

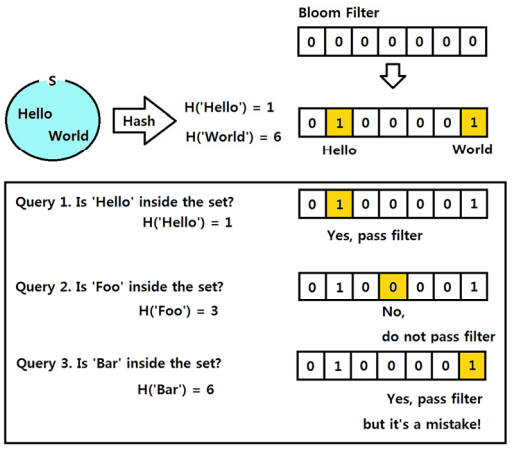

그림 1 블룸 필터
Fig. 1 Bloom Filter

필터의 생성을 위해서 집합 S 안에 있는 Hello와 World 원소에 대해서 해시 함수로 얻은 값을 인덱스로 사용하여 1로 갱신하는 과정을 볼 수 있다. 블룸 필터의 생성이 끝난 후에는 질의를 할 수 있게 된다.

Query 1은 Hello 원소가 집합 S에 존재하는지 여부에 대해서 묻는다. Hello 원소에 대한 해시 함수의 인덱스에 저장된 값이 1이기 때문에 이는 Hello가 집합에 포함됨을 의미한다. 또한, 집합 S에 Hello라는 원소가 포함되어 있음을 확인할 수 있다. 반면에, Query 2는 Foo 원소에 대한 해시 함수의 인덱스 안에 저장된 값이 0이라는 값을 갖기 때문에 Foo라는 원소가 집합에 포함되지 않음을 의미한다. 또한, 집합 S에 Foo라는 원소가 포함되지 않음을 확인할 수 있다. 마지막으로 Query 3을 살펴보면 Bar라는 원소가 집합 S에 포함되어 있지 않지만, 포함되었다는 출력을 갖는다. 위와 같은 경우를 거짓 양성이라고 부른다. 블룸 필터에 거짓 양성이 발생함에도 불구하고, 집합 S를 표현하는데 적은 양의 공간을 사용하기 때문에 여러 분야에서 활발하게 사용되고 있다.

예를 들어, Monkey[5]는 LSM 트리 기반의 키-값 데이터베이스이다. Monkey는 키-값의 존재 여부를 효율적으로 확인하기 위해서 블룸 필터를 사용한다. LSM 트리에 블룸 필터를 사용하지 않았다면, 각 키-값을 조회하기 위해서 디스크 I/O가 발생했을 것이다. 하지만, 블룸 필터가 양성을 반환했을 때만 디스크 I/O를 수행하여 전체적인 데이터베이스의 성능을 개선시킨다.

CRLite[6]의 경우에는 HTTPS 지원을 위한 기관 인증서 정보를 계층적인 블룸 필터를 통해서 사용할 것을 제안한다. CRLite의 사용으로 여러 개의 인증서 정보를 하나의 계층적인 블룸 필터로 표현하여, 공간 효율적으로 인증서 정보를 확인할 수 있게 된다.

쿠쿠 필터[2]는 2013년 B. Fan에 의해서 제시된 블룸 필터의 변형이다. 쿠쿠 필터는 쿠쿠 해싱 기법을 이용해서 집합을 표현하는 자료구조이다. 쿠쿠 필터는 3% 이하의 거짓 양성 확률인 경우에, 블룸 필터보다 더 적은 공간을 사용하여 동일한 거짓 양성 비율을 가질 수 있다. 따라서, 쿠쿠 필터는 3% 이하의 거짓 양성 확률을 갖는 블룸 필터 대신에 사용할 수 있는 자료구조로 볼 수 있다. 기존의 해싱 기법은 충돌이 일어나면, 거짓 양성이 발생한다. 반면에, 쿠쿠 해싱 기법을 이용하면 충돌이 일어난 경우 이전에 위치하고 있던 원소를 다른 인덱스로 옮겨서 거짓 양성이 발생하지 않도록 한다.

XOR 필터[3]는 2019년 T. M. Graf에 의해서 제시된 블룸 필터의 변형이다. XOR 필터는 3개의 해시 함수를 이용해서 집합을 표현하는 자료구조이다. 이는 블룸 필터 그리고 쿠쿠 필터보다 개선된 거짓 양성 비율을 갖는다. 따라서, 집합의 크기가 정적인 경우에는 블룸 필터 또는 쿠쿠 필터 대신에 사용할 수 있는 자료구조이다. 3개의 해시 함수를 사용하는 이유는 비순환 3-분법 랜덤 하이퍼그래프를 생성하기 위함이다. 즉, 3-분법 그래프의 특성을 사용하여 3개의 해시 함수에 XOR 연산을 수행하면 데이터에 대한 집합의 포함 유무를 반환할 수 있게 된다.

학습 블룸 필터[1]는 2018년 T. Kraska에 의해서 제시된 블룸 필터의 변형이다. 학습 블룸 필터는 집합의 포함 여부 문제를 데이터 과학 문제로 해석하여, 이진 분류 모델을 전처리 과정에 사용할 것을 제안한다. 학습 블룸 필터는 모델, 임계치 그리고 보조 필터로 구성된다. 전처리 과정에 사용되는 모델은 [0, 1] 사이의 값을 반환한다. 해당 값이 임계치보다 크면 집합에 포함됨을 의미하고 임계치보다 같거나 작으면 집합에 포함되지 않음을 의미한다. 이후에 모델이 분류하지 못한 원소들은 보조 필터에 의해서 집합의 포함 여부를 반환하게 된다. 전처리 과정인 모델에서는 거짓 음성을 반환될 수 있지만, 블룸 필터와 동일한 역할을 수행하는 보조 필터에 의해서 거짓 음성이 발생하지 않게 된다.

다음은 학습 블룸 필터에 대한 예시를 살펴보도록 한다. 그림 2의 Query 1은 모델만을 이용해서 원소가 분류되는 경우이다. Hello 원소는 기계학습 모델에 의해서 양성으로 분류되었으며, 집합 S 안에서도 Hello 원소가 존재함을 확인할 수 있다. Query 2는 모델이 원소를 음성으로 분류하여, 보조 필터를 확인하는 경우이다. World라는 원소는 모델에 의해서 분류되지 못했지만, 보조로 사용되는 블룸 필터에 의해서 양성을 반환하게 된다. 또한, 집합 S 안에서 World 원소가 존재함을 확인할 수 있다. Query 3의 경우에는 모델과 보조 필터 모두 음성을 반환한 경우이다. 위 경우에는 해당 원소가 집합

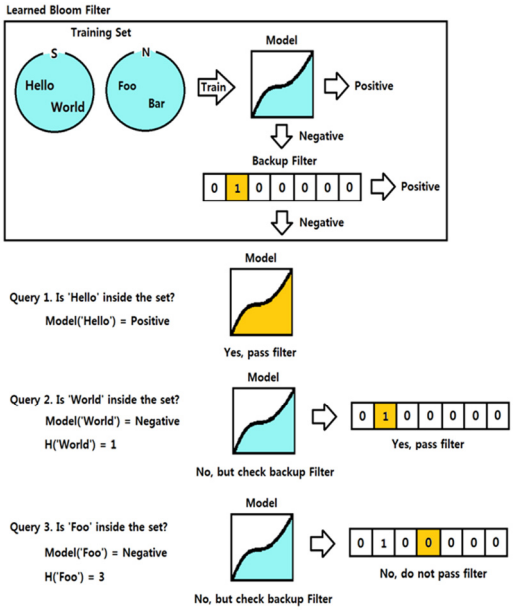

그림 2 학습 블룸 필터

Fig. 2 Learned Bloom Filter

S에 존재하지 않는 경우이다. 집합 S에 Foo라는 원소가 없음을 확인할 수 있으며, 기계학습을 위해서 Foo라는 원소가 음성 데이터 집합 N에서 사용된 것을 확인할 수 있다.

학습 해시 함수[1]는 2018년 T. Kraska에 의해서 제안된 해시 함수이다. 이는 모델의 출력값인 [0, 1] 사이의 값을 비트 벡터의 크기인 m으로 곱하여, 모델의 출력값을 비트 벡터의 인덱스로 매핑하는 방법이다. 즉, 모델을 f 그리고 보조 블룸 필터의 크기를 m으로 두면 $\lfloor f(x) * m \rfloor$을 통해서 비트 벡터의 인덱스를 계산할 수 있다. 여기서 x는 블룸 필터의 입력 또는 질의하는 원소를 의미한다. 일반적인 해시 함수는 원소를 균일하게 분배해서 성능을 개선한다면, 학습 해시 함수는 동일하게 분류된 원소간의 충돌을 늘려서 성능을 개선하였다.

Adaptive Learned Bloom Filter[7]에서는 Ada-BF와 Disjoint Ada-BF라는 새로운 자료구조를 소개한다. Ada-BF의 경우에 데이터의 분포에 따라서 해시 함수의 개수를 변경하여 거짓 양성 비율을 개선한다. 학습 블룸 필터의 전처리 과정에 위치한 모델의 출력값은 [0, 1] 사이로 해당 값이 높은 경우는 원소가 존재할 확률이 높다는 의미이며, 해당 값이 낮은 경우에는 원소가 존재할 확률이 낮다는 의미이다. 따라서 원소가 집합에 존재할 확률이 높은 경우에는 하나의 해시 함수를 사용하고, 원소가 집합에 존재할 확률이 낮은 경우에는 여러 개의 해시 함수를 사용하여 원소의 포함 여부를 표현한다.

Disjoint Ada-BF의 경우에는 데이터 분포를 사용한다는 발상은 동일하지만, 기존에 하나로 사용되던 보조 필터를 여러 개의 보조 필터로 나눠서 사용하는 방법을 제안한다. 즉, 모델에서 반환되는 숫자의 범위인 [0, 1]을 g개의 그룹으로 균일하게 나누고 그룹에 속하는 원소가 많은 경우에는 크기가 큰 보조 필터를 사용하고, 그룹에 속하는 원소가 적은 경우에는 크기가 작은 보조 필터를 사용한다. 위의 방법은 m으로 제한된 비트 벡터의 공간을 효율적으로 사용하여, 거짓 양성 비율을 개선하는 방법이다. 단, 위에서 설명한 g개의 그룹과 k개의 해시 함수의 숫자는 휴리스틱한 방법을 사용하여 설정한다. 따라서, 각 초매개변수 g와 k에 대해서는 사용자가 입력한 범위 내에서 직접 Ada-BF 또는 Disjoint Ada-BF를 반복적으로 생성해가며 최적의 거짓 양성 비율을 갖는 인스턴스를 찾는다. 위와 같이 여러 개의 Ada-BF를 생성하고, 최적의 Ada-BF를 찾을 수 있는 이유는 입력될 데이터 분포를 사전에 알고 있기 때문이다. 학습 블룸 필터는 정적인 집합에 대해서만 실용적으로 사용이 가능하여, 학습 블룸 필터의 변형인 Ada-BF와 Disjoint Ada-BF도 위의 가정을 따른다.

Partitioned Learned Bloom Filter[8]는 Disjoint Ada-BF의 후속 연구이다. 이는 초매개변수 g를 설정하는데 사용한 휴리스틱한 방법을 수학적인 방법으로 대체하여 거짓 양성 비율을 개선한다. 위 논문은 g개의 그룹을 나누기 위한 구간과 각 그룹에서 사용되어야 하는 비트 벡터의 크기를 최적화 이론을 사용하여 수식으로 표현한다. 제안된 방법인 Partitioned Learned Bloom Filter를 사용하면 Disjoint Ada-BF에 비해서 개선된 거짓 양성 비율을 가진다.

3. 확장 학습 블룸 필터

기존의 학습 블룸 필터는 보조 필터로 블룸 필터를 단독으로 구성하여 사용한다. 본 논문에서 제안하는 확장 학습 블룸 필터는 보조 필터에 학습 해시 함수를 사용하기 위해서 α라는 초매개변수를 추가한다. 즉, 확장 학습 블룸 필터의 보조 필터는 블룸 필터와 학습 해시 함수를 같이 사용하는 방법으로 볼 수 있다. α라는 초매개변수는 [0, 1] 사이의 숫자를 가지며, 전체 비트 벡터 m 중에 학습 해시 함수가 사용되는 비율을 나타낸다. 따라서, 확장 학습 블룸 필터는 학습 해시 함수의 공간을 $\lceil \alpha * m \rceil$ 사용하고, 블룸 필터로 사용하는 공간을 $\lfloor (1-\alpha) * m \rfloor$으로 사용한다. 이는 $\lceil \alpha * m \rceil$ 크기의 비트 벡터(B_L)는 학습 해시 함수를 사용하고 $\lfloor (1-\alpha) * m \rfloor$ 크기의 비트 벡터(B_B)는 일반적인 해시 함수가 사용되는 것을 의미한다.

다음은 확장 학습 블룸 필터의 구조에 대해서 살펴본다. 그림 3을 살펴보면, 학습 블룸 필터에서 단독으로

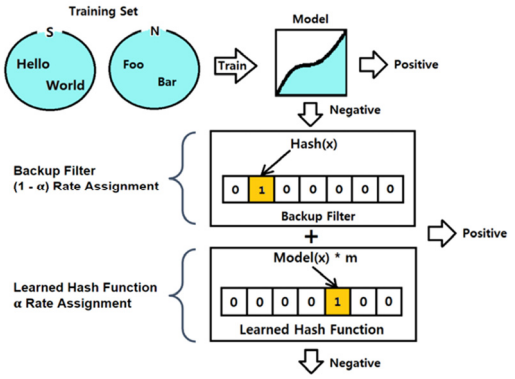

그림 3 확장 학습 블룸 필터
Fig. 3 Extended Learned Bloom Filter

구성되어 있던 보조 필터는 이제 블룸 필터 역할을 수행하는 보조 필터와 학습 해시 함수로 나눠진 것을 확인할 수 있다. 블룸 필터가 해시 함수를 사용해서 원소들은 균일하게 비트 벡터 공간에 분산시켜서 사용한다면, 학습 해시 함수는 기계학습 모델을 이용하여 동일한 클래스간의 충돌을 늘린다. 따라서, 각자 다른 특징을 가진 해시 함수를 함께 사용하므로 기존의 블룸 필터보다 개선된 거짓 양성 비율을 가질 수 있게 된다.

다음은 확장 학습 블룸 필터의 예시에 대해서 살펴본다. 그림 4를 보면, Query 1은 모델에 의해서 양성으로 분류된 경우이다. Hello 원소는 모델에 의해서 분류되었으며, 집합 S 내부에 질의한 원소인 Hello가 존재하는 것을 확인할 수 있다. Query 2의 경우에는 모델이 분류하지 못하였으나, 보조 필터와 학습 해시 함수에 의해서 원소가 양성으로 분류된 경우이다. World 원소는 모델에서 구분하지 못하였지만, 보조 필터와 학습 해시 함수에 모두 인덱스를 1로 가져서 집합에 존재하는 것으로 판별된다. 또한, 집합 S에 질의한 원소인 World가 존재하는 것을 확인할 수 있다. Query 3의 경우에는 학습 해시 함수에서는 존재한다고 판단하였지만, 블룸 필터인 보조 필터가 존재하지 않는다고 판단한 경우이다. 확장 학습 블룸 필터에서는 보조 필터와 학습 해시 함수에서 모두 양성으로 반환해야 최종적으로 양성을 반환하게 된다. 현재는 보조 필터에서 음성을 반환하므로, Foo라는 원소는 집합에 포함되어 있지 않음으로 분류된다. 또한, 집합 S에 Foo 원소가 존재하지 않는 것을 확인할 수 있다. 마지막으로, Query 4는 보조 필터에서 존재한다고 판단하였지만, 학습 해시 함수에서는 존재하지 않는다고 판단한 경우이다. 이 경우에도 학습 해시 함수에서 음성을 반환하므로, Bar 원소는 집합 S에 존재하지 않는다고 판단한다. 또한, 집합 S에 Bar 원소가 존재하지 않는 것을 확인할 수 있다.

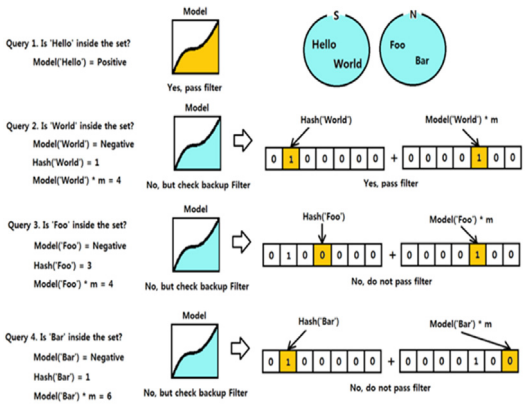

그림 4 확장 학습 블룸 필터의 개요
Fig. 4 Overview of Extended Learned Bloom Filter

블룸 필터는 집합의 크기 n과 거짓 양성 확률 p가 주어지면, 최적의 해시 함수 개수 k와 비트 벡터의 크기 m을 구할 수 있다. 위에서 구한 비트 벡터의 크기 m에 대해서 α 비율만큼의 공간을 학습 해시 함수로 분배한다. 보조 필터의 공간은 α 비율 만큼 줄어들었기 때문에 반환되는 거짓 양성 오류는 분배 이전보다 많아지게 된다. 따라서, 학습 해시 함수가 거짓 양성 비율을 낮추는데 일조하지 않는다면 기존의 학습 블룸 필터보다 개선된 거짓 양성 비율을 갖지 못한다.

다음은 확장 학습 블룸 필터의 수도 코드에 대해서 살펴본다. Algorithm 1은 확장 학습 블룸 필터를 초기화하는 알고리즘이다. 초매개변수 α를 통해서 보조 필터의 공간 m을 학습 해시 함수가 사용되는 공간인 m_L과 블룸 필터가 사용되는 공간인 m_B로 공간을 나누는 것을 확인할 수 있다. 학습 해시 함수와 해시 함수는 동작 방식이 다르기 때문에 동일한 공간에서 함께 사용할 수 없다. 따라서 위와 같이 공간을 나누는 과정을 수행한다.

Algorithm 2는 확장 학습 블룸 필터의 입력 알고리즘이다. 모델값이 임계치보다 큰 값을 가지면, 데이터가 모델에 의해서 저장되었음을 의미한다. 따라서, 별도로 학습 해시 함수와 보조 필터를 사용하지 않아도 된다. 하지만, 모델값이 임계치보다 같거나 작은 값을 가지면, 이는 모델이 데이터를 정상적으로 분류하지 못함을 의미한다. 위와 같은 경우에는 보조 필터인 블룸 필터와 학습 해시 함수를 사용해서 데이터를 저장하는 과정을 가진다. 마지막으로, Algorithm 3은 확장 학습 블룸 필터의 질의 알고리즘이다. 확장 학습 블룸 필터의 입력하는 알고리즘과 유사하며, 각 해시 함수에 대한 값 중에 하나라도 0이라는 값이 나오면 음성을 반환하고 종료한다. 단, 모든 해시 함수의 값이 1이라면 양성을 반환하는 알고리즘이다.

Algorithm 1 Initialization of Extended LBF

```
1. Function init(α, m):
2.     m_L = ⌈ α * m ⌉
3.     m_B = ⌊ (1 - α) * m ⌋
4.     B_L = bitarray(m_L)
5.     B_B = bitarray(m_B)
6. end
```

Algorithm 2 Insert of Extended LBF

```
1. Function insert(x):
2.     model_value = f(x)
2.     if model_value > threshold:
3.         pass
4.     else:
5.         for i = 1, ... , k:
6.             B_B [h_i(x)] = 1
7.         end
8.         B_L[model_value * m_L] = 1
9.     end
10. end
```

Algorithm 3 Query of Extended LBF

```
1. Function query(x):
2.     model_value = f(x)
2.     if model_value > threshold:
3.         return True
4.     else:
5.         for i = 1, ... , k:
6.             if B_B[h_i(x)] == 0:
7.                 return False
8.             end
9.         end
10.        if B_L[model_value * m_L] == 0:
11.            return False
12.        end
13.    end
14.    return True
15. end
```

3.1 거짓 양성 보장

확장 학습 블룸 필터가 블룸 필터와 동일한 기능을 수행하기 위해서는 거짓 음성이 반환되지 않아야 한다. 이를 증명하기 위해서 확장 학습 블룸 필터에 거짓 음성이 발생했다고 가정해본다. 거짓 음성이 발생했다는 것은 양성 데이터에 대해서 음성의 결과가 반환된 것이다. 확장 학습 블룸 필터의 전처리 과정으로 사용되는 모델은 양성만을 반환하므로, 보조 필터에서 음성이 반환되어야 한다. 하지만, 보조 필터는 해시 함수 또는 학습 해시 함수를 사용하여 모든 원소에 대해서 입력을 수행하기 때문에 거짓 음성을 반환할 수 없다. 위의 모순에 의해서 확장 학습 블룸 필터는 거짓 음성이 발생하지 않는다는 것을 알 수 있다.

3.2 공간 분석

모델의 크기를 model, 전체 보조 필터의 크기를 m, $\lfloor (1-\alpha) * m \rfloor$ 크기의 보조 필터를 m_B, $\lceil \alpha * m \rceil$ 크기의 학습 해시 함수 공간을 m_L이라고 가정한다. 추가적으로 P, P_B, P_L은 각각 전체 보조 필터, 나눠진 보조 필터 그리고 학습 해시 함수에 대한 거짓 양성 확률이다. 확장 학습 블룸 필터가 학습 블룸 필터와 같거나 적은 공간을 사용하기 위해서는 다음의 조건을 만족해야 한다.

$$m_L + m_B + model \leq m + model$$

$$m_L + m_B \leq m$$

$$m_L + \frac{-nlnP_B}{(ln2)^2} \leq \frac{-nlnP}{(ln2)^2}$$

$$m_L + \frac{-nlnP_B}{(ln2)^2} \leq \frac{-nlnP_B P_L}{(ln2)^2}$$

$$m_L \leq \frac{-nlnP_L}{(ln2)^2}$$

m과 m_B는 보조 필터로 이를 블룸 필터의 비트 벡터 크기를 구하는 공식으로 대체할 수 있다. 또한, 학습 해시 함수와 보조 필터는 함께 사용되어 원소를 판별하기 때문에 $P = P_B * P_L$ 공식을 사용하여 위의 수식을 정리할 수 있다. 결과적으로 학습 해시 함수에 사용되는 공간인 m_L은 P_L의 거짓 양성 확률을 가지는 블룸 필터보다 같거나 작아야 성능이 개선된다. 즉, 학습 해시 함수가 블룸 필터보다 적은 공간을 사용하면서 동일한 거짓 양성 비율을 가지게 된다면 확장 학습 블룸 필터는 학습 블룸 필터에 비해서 공간을 적게 사용할 수 있다.

3.3 격자 탐색

학습 해시 함수는 데이터와 기계학습 모델의 성능에 의해서 거짓 양성 비율이 결정된다. 따라서, 확장 학습 블룸 필터에서는 학습 해시 함수의 성능이 좋은 경우 학습 해시 함수를 사용하는 공간을 늘리고, 학습 해시 함수의 성능이 좋지 않은 경우 학습 해시 함수를 사용하는 공간을 줄이기 위해서 격자 탐색을 수행한다.

다음은 격자 탐색 알고리즘에 대해서 알아본다. Algorithm 4에서는 보조 필터로 사용될 공간 m이 입력으로 주어진다. 이후에, 학습 해시 함수의 비율을 나타내는 초매게변수 α의 값을 0.01에서 0.50까지 늘려나가면서 가장 낮은 거짓 양성 비율을 갖는 확장 학습 블룸 필터를 탐색해 나간다. 이는 보조 필터인 블룸 필터와 학습 해시 함수가 함께 사용되는 경우에 성능이 가장 좋은 공간의 비율을 찾기 위한 과정으로 볼 수 있다. 또한, 거짓 양성 비율을 측정하기 위해서 음성 데이터로만 구성된 테스트 데이터를 사용한다.

Algorithm 4 Grid Search of Extended LBF

```
1. Function grid_search(m):
2.     best_model = None
3.     min_FPR = ∞
4.     for α = 0.01, ···, 0.50:
5.         init(α, m)
6.         for data in train_data:
7.             insert(data)
8.         end
9.         FP_count = 0
10.        for data in negative_test_data:
11.            FP_count += query(data)
12.        end
13.        FPR = FP_count / len(negative_test_data)
14.        if FPR < min_FPR:
15.            min_FPR = FPR
16.            best_model = α
17.        end
18.    end
19. end
```

4. 구현 사항

다음은 확장 학습 블룸 필터를 구현하며 발생했던 문제와 개선점에 대해서 알아본다. 구현 사항과 관련된 실험 결과는 5장에서 별도로 설명하도록 한다.

4.1 모델 탐색

기계학습 모델도 여러 번의 학습을 통해서 최종적으로 가장 좋은 모델을 선택하듯이, 확장 학습 블룸 필터에서도 성능이 좋은 모델을 탐색하는 과정을 필요로 한다. 예를 들어, 비트 벡터의 크기 m이 10,000이라면 학습 해시 함수가 사용되는 공간을 1비트씩 증가하면서 확장 학습 블룸 필터의 성능을 측정해볼 수 있다. 하지만, 확장 학습 블룸 필터를 1비트씩 탐색하면 총 10,000회의 탐색이 필요하며, 이는 시간과 비용이 많이 든다는 단점이 있다. 반면에 확장 학습 블룸 필터의 변수 α를 0.01씩 증가하면서 성능을 측정한다면, 100회 탐색으로 성능이 좋은 확장 학습 블룸 필터를 찾을 수 있다. 결과적으로 격자 탐색을 통해서 성능이 좋은 확장 학습 블룸 필터를 찾는데 드는 시간과 비용을 절약할 수 있다.

4.2 모델의 정밀도

모델의 정밀도 문제는 학습 해시 함수에 입력을 하였으나, 질의하였을 때는 입력된 원소에 대해서 인식하지 못하는 문제이다. 수학적으로는 [0, 1] 사이에 무수히 많은 실수가 있겠지만, 공학적으로는 [0, 1] 사이에 있는 수를 모두 표현할 수 없다. 따라서, 이를 32비트 또는 64비트 부동소수점으로 표현하며 [0, 1] 사이의 일부 숫자만 정확히 표현이 가능하다. 확장 학습 블룸 필터를 생성할 때는 인공신경망의 배치 크기를 늘려서 생성에 걸리는 시간을 줄일 수 있다. 하지만, 확장 학습 블룸 필터가 생성된 후에는 질의가 오면 바로 응답하기 위해서 배치 크기를 1로 사용한다. 생성과 질의를 할 때의 배치 크기 차이로 인해서 동일한 원소임에도 불구하고 인공신경망이 서로 다른 출력값을 반환하는 경우가 있다. 즉, 부동소수점 오류로 인하여 학습 해시 함수가 결정론적 특성을 잃어버리는 경우가 발생할 수 있음을 보인다. 모델의 정밀도 문제는 인공신경망 내부에 사용되던 32비트 부동소수점을 64비트 부동소수점으로 변경하면 해결할 수 있다.

4.3 모델 조정

학습 블룸 필터[1]를 제안한 T. Kraska는 모델로 글자 임베딩[9] 32차원, GRU[10] 16차원 그리고 완전 연결 계층 1차원을 사용할 것을 제안한다. 본 논문에서는 인공신경망의 구조를 글자 임베딩 16차원, GRU 16차원, 완전 연결 계층 8차원 그리고 완전 연결 계층 1차원으로 인공신경망 모델을 조정한다. 그림 5를 통해서 T. Kraska가 제안한 모델과 본 논문에서 제안한 모델의 인공신경망 구조를 살펴볼 수 있다. 모델 조정를 통해서 기존의 4,017개의 변수로 이루어진 인공신경망을 2,577개로 줄임과 동시에 거짓 양성 비율에 대해서는 개선된 성능을 갖는 것을 보인다. 즉, 인공신경망에 사용되는 메모리를 40KB에서 20KB로 절약하고, 절약된 20KB의 공간을 보조 필터에 사용하는 방법이라고 볼 수 있다.

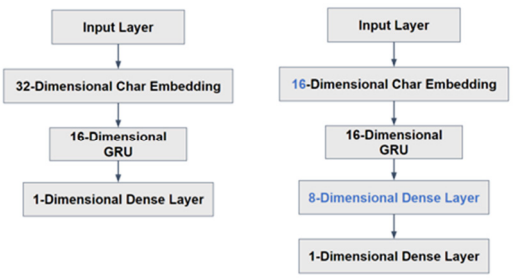

그림 5 모델 조정
Fig. 5 Model Adjustment

4.4 학습 해시 함수의 이해

확장 학습 블룸 필터는 전처리 과정에서 사용하던 모델을 보조 필터에서 학습 해시 함수로 재사용하게 된다. 학습 해시 함수의 사용으로 확장 학습 블룸 필터의 거짓 양성 비율이 감소하는 것은 각 해시 함수의 특성이 다르기 때문이다. 기존의 해시 함수는 원소를 균일하게 퍼트려서 비트 벡터를 사용하는 특성을 가지고 있다. 반면에, 학습 해시 함수의 경우에는 동일한 클래스간의 충돌을 최대화해서 공간 사용을 효율적으로 한다.

학습 해시 함수의 기계학습 모델은 이진 교차 엔트로피(Cross Entropy)를 통해서 학습된다. 학습 데이터가 분포 p를 가지고 있고, 기계학습 모델로 예측된 데이터가 분포 q를 가진다고 가정한다. 분포 p와 q에 대한 이진 교차 엔트로피 수식을 정리하면, 이는 쿨백-라이블러 발산 최소화하는 수식으로 나타낼 수 있다. 쿨백-라이블러 발산은 분포 간의 차이를 나타내는 지표로, 기계학습 모델의 학습이 정상적으로 수행된다면 학습 데이터와 유사한 분포를 가질 수 있음을 나타낸다. 즉, 기계학습 모델로 예측된 값을 사용하면 동일한 클래스간의 충돌을 늘릴 수 있음을 의미한다.

$$H(p,q) = -\sum_{x \in \chi} p(x) log q(x)$$
$$= H(p) + D_{KL}(p \| q)$$

5. 실 험

실험은 Intel Core i5 4690 @ 3.50Ghz CPU와 8GB Samsung DDR3 RAM 2개를 사용하는 환경에서 진행되었다. 사용한 소프트웨어로는 Conda 4.10.1, Python 3.7.4, Tensorflow 2.4 그리고 Scikit-learn 0.24.1 버전을 사용하였다. 위 실험 환경을 바탕으로 모델 탐색, 모델의 정밀도 그리고 모델 조정에 대한 실험 결과에 대해서 살펴본다.

5.1 데이터

인공신경망을 학습하기 위한 데이터는 Shalla's Blacklist [11]를 사용한다. Shalla's Blacklist는 금융, 자동차 등의 81개의 카테고리로 구성되어 있다. 각 카테고리별로 URL정보가 존재하며 특정 카테고리의 접속을 방지하기 위한 데이터이다. 확장 학습 블룸 필터를 실험하기 위해서 성인용으로 구분된 URL 데이터 1,491,178개와 그 외의 분야로 구분된 URL 데이터 1,435,527개를 사용한다. 위 데이터를 학습에 사용할 때는 8:1:1 비율을 각각 학습, 개발, 테스트 데이터로 사용한다. 학습 데이터는 인공신경망을 학습하기 위해서 사용하고, 개발 데이터는 학습 블룸 필터의 임계치를 계산하기 위해서 사용된다. 마지막으로, 테스트 데이터는 모델의 성능을 평가할 때 사용된다.

5.2 모델

모든 실험은 본 논문에서 제안한 글자 임베딩 16차원, GRU 16차원, 완전 연결 계층 8차원 그리고 완전 연결 계층 1차원으로 구성된 확장 학습 블룸 필터를 기준으로 한다. 위 인공신경망에 대한 입력 계층은 50차원을 사용하고 이는 URL 50글자를 읽어서 사용하는 것을 의미한다.

글자 임베딩은 글자를 벡터로 변환해주는 역할을 수행한다. 글자 임베딩은 glove.6B.50d로 미리 학습된 단어 임베딩 데이터를 사용해서 만들 수 있다. 위의 glove.6B.50d는 단어 임베딩에 60억개의 단어를 50차원의 벡터로 구성하였음을 의미한다. 단어 임베딩을 글자 임베딩으로 바꾸기 위해서는 각 글자가 사용되는 모든 단어 벡터에 평균을 취한다. 위의 연산으로 얻어진 글자 임베딩은 68개의 글자에 대해서 50차원의 벡터를 가진다. 단, 인공신경망은 16차원의 글자 임베딩을 사용하므로 위에서 구한 50차원의 글자 임베딩을 주성분 분석(PCA)을 통해서 16차원으로 축소해서 사용한다.

GRU는 자연어 처리를 위해서 사용된 계층이다. 다음으로 사용된 완전 연결 계층의 경우에는 모델의 정확성을 높이기 위해서 추가한 계층이며, ReLu 활성화 함수를 사용한다. 출력 계층은 [0, 1] 사이의 숫자를 출력하기 위해서 시그모이드(Sigmoid) 활성화 함수를 사용한다.

인공신경망의 손실 함수는 교차 엔트로피를 사용한다. 이는 이진 분류를 할 때 사용되며, 블룸 필터가 해결하려고 하는 문제와 동일다고 볼 수 있다. 인공신경망의 학습과 관련된 초매개변수는 학습률 0.005, 배치 크기 1024 그리고 에포크는 40으로 설정한 상태로 실험을 진행한다. 단, 학습이 끝난 후에는 인공신경망의 배치 크기를 1로 사용한다.

5.3 모델 탐색 실험

모델 탐색 실험은 확장 학습 블룸 필터의 α 변수를 0.01에서 0.50까지 변경하면서 거짓 양성 비율을 블룸 필터 그리고 학습 블룸 필터와 비교한다. 단, 전처리 과정에 사용되는 인공신경망 구조는 모두 동일하며 데이터의 개수를 1,000개, 10,000개, 100,000개, 1,000,000개 사용한다.

그림 6인 모델 탐색에 대한 실험 결과를 보면 1,000개와 10,000개의 데이터가 사용된 경우에는 확장 학습 블룸 필터와 학습 블룸 필터는 블룸 필터에 비해서 좋지 않은 성능을 가지는 것을 살펴볼 수 있다. 이는 인공신경망의 크기가 수용해야 하는 데이터의 개수보다 크기

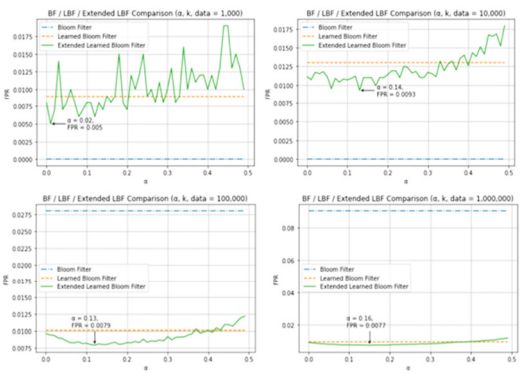

그림 6 모델 탐색 실험 결과
Fig. 6 Model Navigation Experimental Results

때문에 일어나는 현상이다. 또한, 학습 데이터가 많지 않아서 확장 학습 블룸 필터의 거짓 양성 비율 그래프에 노이즈가 발생하는 모습을 볼 수 있다.

반면에 데이터를 100,000개와 1,000,000개 사용하는 경우에는 확장 학습 블룸 필터와 학습 블룸 필터가 블룸 필터에 비해서 20% 개선된 거짓 양성 비율을 갖는 것을 확인할 수 있다. 또한, 학습 데이터가 많아져서 확장 학습 블룸 필터의 거짓 양성 비율에 발생하는 노이즈가 줄어든 것을 확인할 수 있다. 그림 6의 모델 탐색 실험 결과를 보면 학습 데이터가 1,000개 사용된 경우 거짓 양성 비율이 일정하지 않지만, 학습 데이터가 1,000,000개 사용된 경우에는 거짓 양성 비율이 일정한 것을 확인할 수 있다. 즉, 학습 해시 함수를 사용하기 위해서는 일정량의 학습 데이터가 주어져야 한다는 것을 실험적으로 보인다.

5.4 모델의 정밀도 실험

모델의 정밀도 실험은 학습 해시 함수를 사용하는 환경에서 배치 크기를 조정하는 경우 거짓 음성 오류가 발생할 수 있음을 실험적으로 보인다. 모델의 정밀도 실험은 확장 학습 블룸 필터의 α 변수를 0.99로 설정한 상태에서, 보조 필터의 크기를 강제적으로 10KB, 100KB, …, 10GB의 공간을 사용하도록 설정한다. 또한, 32와 64비트 부동소수점 인공신경망을 100,000개의 데이터로 학습시켜서 확장 학습 블룸 필터의 참 양성 비율을 확인한다.

그림 7인 모델의 정밀도 실험 결과를 보면 32비트 부동소수점 인공신경망을 사용하는 확장 학습 블룸 필터의 경우 10KB부터 거짓 음성 오류가 발생하는 것을 확인할 수 있다. 반면에, 64비트 부동소수점 인공신경망을 사용하는 확장 학습 블룸 필터의 경우에는 10GB의 보조 필터 크기에서도 오류가 발생하지 않는 것을 확인할 수 있다. 따라서, 학습 해시 함수를 사용하는 공간이 10KB 이상인 경우에는 64비트 부동소수점의 인공신경망을 사용해야 거짓 음성 오류가 발생하지 않는 것을 확인할 수 있다.

64 비트 부동 소수점의 경우에는 15자릿수까지 유효하다. 따라서, 이는 보조 필터의 크기가 약 100TB인 공간까지 오류를 발생시키지 않는다는 것을 의미한다. 블룸 필터의 거짓 양성 확률 p가 0.01인 상태에서 100TB 크기의 블룸 필터를 생성하기 위해서는 집합의 크기 n이 100조의 크기를 가져야 한다. 블룸 필터를 생성하는 대부분의 경우에 원소의 개수가 100조 보다 작기 때문에 모델의 정밀도 문제는 발생하지 않을 것이다.

5.5 모델 조정 실험

모델 조정 실험은 2번의 실험을 통해서 이루어진다. 첫번째 실험은 블룸 필터, 쿠쿠 필터, XOR 필터, 학습 블룸 필터 그리고 확장 학습 블룸 필터의 질의 시간을 비교한다. 실험은 각 블룸 필터에 대해서 질의를 10,000회 수행하는 방식으로 진행된다. 질의에 사용된 데이터는 블룸 필터에 입력된 데이터와 입력되지 않은 데이터에 대해서 동일한 실험을 총 10회씩 진행하고, 10회에 대한 평균 값을 각 블룸 필터의 질의시간으로 사용하였다. T. Kraska가 제안한 모델은 32차원의 글자 임베딩을 사용하므로 접미사로 32를 사용하며, 본 논문에서 제안한 모델은 16차원의 글자 임베딩을 사용하므로 접미사로 16을 사용한다.

그림 8인 모델 조정에 대한 시간 측정 실험 결과를 보면 학습 블룸 필터와 확장 학습 블룸 필터의 질의시간은 각각 278초와 281초로 서로 유사한 것을 확인할 수 있다. 즉, 본 논문에서 제안한 인공신경망의 구조가 기존의 T. Kraska가 제안한 모델에 비해서 3초 정도 느린 것을 확인할 수 있다. 이는 질의 시간이 2% 느려진 것으로 어플리케이션에 따라서, 학습 블룸 필터 대신에 확장 학습 블룸 필터를 사용할 수 있음을 의미한다. 마지막으로 학습 계열의 블룸 필터는 블룸 필터에 비해서 10,000배 정도 느린 성능을 갖는데, 이는 학습 계열

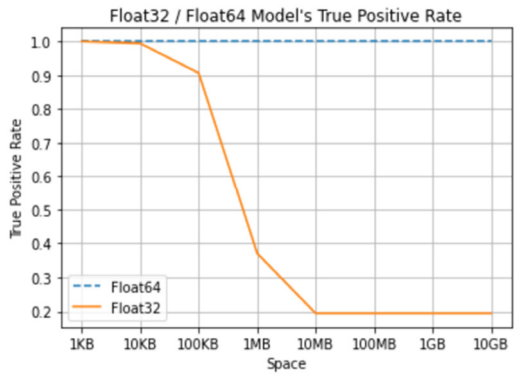

그림 7 모델의 정밀도 실험 결과
Fig. 7 Model Precision Experimental Results

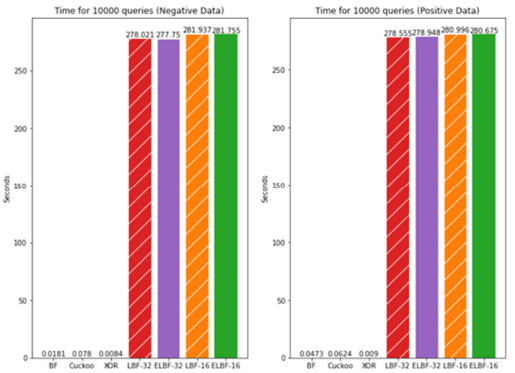

그림 8 모델 조정에 대한 시간 측정
Fig. 8 Time Measurement for Model Tuning

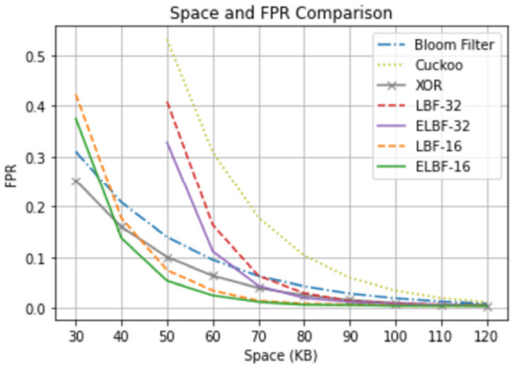

그림 9 모델 조정에 대한 거짓 양성 비율
Fig. 9 False Positive Rate for Model Tuning

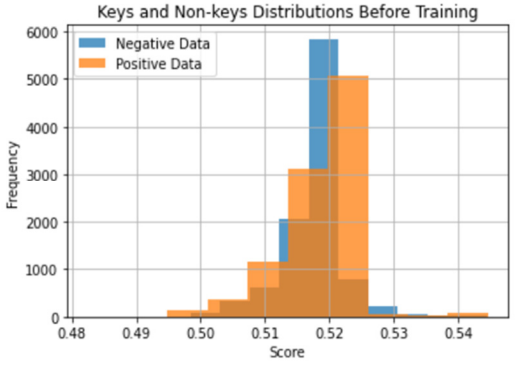

그림 10 학습 이전의 모델 데이터 분포
Fig. 10 Data Distribution of non-trained model

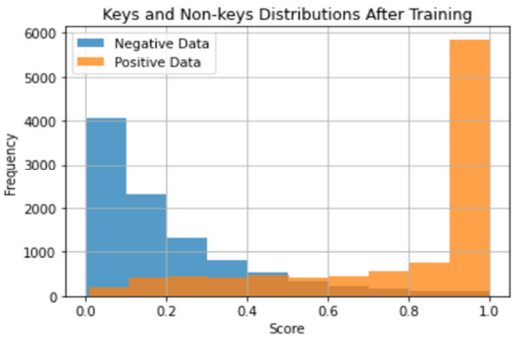

그림 11 학습 이후의 모델 데이터 분포
Fig. 11 Data Distribution of trained model

블룸 필터가 인공신경망을 사용하여 공간적으로 효율적인 대신에 갖는 약점이라고 볼 수 있다.

두번째 실험은 동일한 공간이 주어진 경우 블룸 필터, 쿠쿠 필터, XOR 필터, 학습 블룸 필터 그리고 확장 학습 블룸 필터의 거짓 양성 비율에 대해서 비교한다. 실험에 사용되는 공간은 30KB부터 120KB의 공간을 사용하였으며, 학습 계열 블룸 필터에 사용되는 인공신경망은 100,000개의 데이터를 사용하여 학습하였다.

그림 9인 모델 조정에 대한 거짓 양성 비율 실험 결과를 보면, 30KB에서는 XOR 필터가 가장 좋은 거짓 양성 비율을 갖는다. 하지만, 40KB~110KB까지는 본 논문에서 제안한 인공신경망을 사용하는 확장 학습 블룸 필터가 가장 좋은 거짓 양성 비율을 갖는 것을 확인할 수 있다. 추가적으로 T. Kraska가 제안한 인공신경망을 사용했을 때는 인공신경망의 크기로 인해서 40KB 이전에서는 측정이 불가능하였으며, 이는 본 논문에서 제안한 확장 학습 블룸 필터에 비해서 좋지 않은 거짓 양성 비율을 가진다.

확장 학습 블룸 필터의 보조 필터의 구현체를 바꾸면 추가적으로 거짓 양성 비율을 개선할 수 있다. 실험에서 사용하는 보조 필터는 블룸 필터이다. 블룸 필터와 확장 학습 블룸 필터를 비교하면, 30KB를 제외하고는 확장 학습 블룸 필터의 거짓 양성 비율이 블룸 필터보다 개선되었다. XOR 필터와 쿠쿠 필터는 블룸 필터보다 개선된 거짓 양성 비율을 갖는 자료구조이다. 따라서, 보조 필터에 블룸 필터 대신에 XOR 필터 또는 쿠쿠 필터를 사용하면 확장 학습 블룸 필터의 거짓 양성 비율을 추가적으로 개선할 수 있다.

5.6 학습 해시 함수의 이해 실험

학습 해시 함수의 이해 실험은 기계학습 모델의 학습 전과 학습 이후의 데이터 분포에 대해서 살펴보는 실험이다. 블룸 필터에 입력될 데이터인 양성 데이터와 블룸 필터에 입력되지 않을 데이터인 음성 데이터에 대해서 분포가 겹치지 않는 경우에 해시 함수의 거짓 양성 비율이 개선될 수 있다. 이는 양성 데이터와 음성 데이터가 동일한 인덱스로 매핑되면 거짓 양성이 발생하기 때문이다.

그림 10인 학습 이전의 모델 데이터 분포에 대한 실험 결과를 보면, 모델이 학습하기 이전에는 대부분 0.52 정도의 출력값이 모델에서 나오는 것을 볼 수 있다. 모델의 출력값이 0에 가까울수록 음성 데이터일 확률이 높으며, 반대로 모델의 출력값이 1에 가까울수록 양성 데이터일 확률이 높다. 학습 이전의 기계학습 모델은 0.52 정도의 출력값으로 예측을 랜덤에 가깝게 수행하는 것을 볼 수 있다. 또한, 양성 데이터와 음성 데이터간의 분포에 겹치는 부분이 많다. 이는 거짓 양성이 발생함을 의미한다.

그림 11인 학습 이후의 모델 데이터 분포에 대한 실험 결과를 보면, 모델이 학습한 이후에는 음성 데이터는 대부분 출력값을 0으로 갖고, 양성 데이터는 대부분 출력값을 1로 갖는 것을 확인할 수 있다. 각 양성과 음성 데이터에 대한 분포가 겹친 부분이 많을수록 거짓 양성 현상이 빈번하게 일어나기 때문에, 학습 이후에는 거짓

양성 비율이 줄어들 것임을 직관적으로 이해할 수 있다. 즉, 동일한 인덱스에 양성과 음성 데이터가 매핑되는 경우가 학습 이후에는 줄어들게 되어서, 거짓 양성 비율이 개선될 수 있다.

6. 결 론

본 논문은 학습 블룸 필터의 변형인 확장 학습 블룸 필터를 제안한다. 확장 학습 블룸 필터는 보조 필터의 공간 일부를 학습 해시 함수로 사용하여 거짓 양성 비율을 개선한다. 이는 학습 해시 함수의 비율을 나타내는 초매개변수 α에 격자 탐색을 수행하여 성능이 가장 좋은 확장 학습 블룸 필터를 선택하는 방법으로 거짓 양성 비율을 개선한다. 실험을 통해서 학습 블룸 필터와 동일한 공간을 사용하는 확장 학습 블룸 필터가 20% 이상의 거짓 양성 비율이 개선됨을 실험을 통해서 확인하였다. 또한, 학습 해시 함수를 사용함으로 발생할 수 있는 모델의 정밀도 문제에 대해서 소개하며, 이를 64비트 부동소수점을 사용하여 해결할 것을 제안한다. 마지막으로, 인공신경망의 성능과 확장 학습 블룸 필터의 성능이 꼭 비례하지 않음을 보였다. 인공신경망에 사용되는 공간을 보조 필터에 사용하면, 확장 학습 블룸 필터 관점에서는 20KB의 공간을 절약한 상태에서 동일한 거짓 양성 비율을 가질 수 있음을 실험을 통해서 확인하였다.

확장 학습 블룸 필터는 전처리 과정에 사용되는 인공신경망에 의해서 질의 시간이 느리다는 단점이 있다. 인공신경망 외의 다른 경량화된 기계학습 모델을 적용한다면 질의 시간이 개선될 수 있을 것이다. 또한, 학습 해시 함수에서 모델의 출력값 f(x)에 비트 벡터의 크기 m을 곱하는 것은 공간을 균일하게 사용한다는 것을 의미한다. 모델의 출력값 f(x) 중에 오류가 많이 발생하는 구간에 가중치를 주는 수식을 추가한다면, 학습 해시 함수의 거짓 양성 비율을 추가적으로 개선할 수 있을 것이다.

References

[1] T. Kraska, A. Beutel, E. H. Chi, J. Dean, and N. Polyzotis, "The case for learned index structures," *International Conference on Management of Data (ACM SIGMOD)*, pp. 489-504, Jun. 2018.

[2] B. Fan, D. G. Andersen, M. Kaminsky, and M. D. Mitzenmacher, "Cuckoo filter: Practically better than bloom," *CoNEXT*, pp. 75-88, Dec. 2014.

[3] T. M. Graf and D. Lemire, "Xor Filters: Faster and Samller Than Bloom and Cuckoo Filters," *Journal of Experimental Algorithmics (JEA)*, Vol. 25, No. 1, pp. 1-16, Dec. 2019.

[4] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, Vol. 13, No. 7, pp. 422-426, Jul. 1970.

[5] N. Dayan, M. Athanassoulis, and S. Idreos. Monkey: Optimal Navigable Key-Value Store, *Proc. of the ACM SIGMOD International Conference on Management of Data*, pp. 79-94, May. 2017.

[6] J. Larisch, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson, "CRLite: A Scalable System for Pushing All TLS Revocations to All Browsers," *IEEE Symposium on Security and Privacy*, Jun. 2017.

[7] A. Bhattacharya, B. Srikanta, and B. Amitabha, "Adaptive Learned Bloom Filters under Incremental Workloads," *Proc. of the 7th ACM IKDD CoDS and 25th COMAD*, pp. 107-115, Jan. 2020.

[8] K. Vaidya, E. Knorr and T. Kraska, "Partitioned Learned Bloom Filter," *arXiv preprint arXiv:2006.03176*, 2020.

[9] J. Pennington, R. Socher, and C. D. Manning. "Glove: Global vectors for word representation," *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532-1543, Jan. 2014.

[10] K. Cho, B. Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724-1734, Oct. 2014.

[11] Shalla Secure Services KG [Online]. Available: https://www.shallalist.de (Accessed: 5 July 2021)

양 수 현

2016년 건국대학교 인터넷미디어공학부 학사. 2021년 서울대학교 컴퓨터공학부 석사. 관심분야는 데이터베이스, 학습 인덱스, 블룸 필터

김 형 주

1982년 서울대학교 전산학과 학사. 1985년 Univ. of Texas at Austin 석사. 1988년 Univ. of Texas at Austin 박사. 1988년~1990년 Georgia Institute of Technology 부교수. 1991년~현재 서울대학교 컴퓨터공학부 교수. 관심분야는 데이터베이스, XML, 시맨틱 웹, 빅데이터