



Article

# AR-Sanad 280K: A Novel 280K Artificial Sanads Dataset for Hadith Narrator Disambiguation

Somaia Mahmoud<sup>1</sup>, Omar Saif<sup>2</sup>, Emad Nabil<sup>3,4</sup>, Mohammad Abdeen<sup>3</sup> , Mustafa ElNainay<sup>1,5</sup>   
and Marwan Torki<sup>1,\*</sup>

- <sup>1</sup> Department of Computer and Systems Engineering, Alexandria University, Alexandria 21526, Egypt; es-somaya.mahmoud1415@alexu.edu.eg (S.M.); melnainay@aiu.edu.eg (M.E.)
- <sup>2</sup> Faculty of Hadith and Islamic Studies, Islamic University of Madinah, Madinah 42351, Saudi Arabia; osaif@iu.edu.sa
- <sup>3</sup> Faculty of Computer and Information Systems, Islamic University of Madinah, Madinah 42351, Saudi Arabia; e.nabil@fci-cu.edu.eg (E.N.); mabdeen@iu.edu.sa (M.A.)
- <sup>4</sup> Faculty of Computers and Artificial Intelligence, Cairo University, Giza 12613, Egypt
- <sup>5</sup> Faculty of Computer Science and Engineering, AlAlamein International University, Matrouh 51718, Egypt
- \* Correspondence: mtorki@alexu.edu.eg

**Abstract:** Determining *hadith* authenticity is vitally important in the Islamic religion because *hadiths* record the sayings and actions of Prophet Muhammad (PBUH), and they are the second source of Islamic teachings following the Quran. When authenticating a *hadith*, the reliability of the *hadith* narrators is a big factor that *hadith* scholars consider. However, many narrators share similar names, and the narrators' full names are not usually included in the narration chains of *hadiths*. Thus, first, ambiguous narrators need to be identified. Then, their reliability level can be determined. There are no available datasets that could help address this problem of identifying narrators. Here, we present a new dataset that contains narration chains (*sanads*) with identified narrators. The AR-Sanad 280K dataset has around 280K artificial *sanads* and could be used to identify 18,298 narrators. After creating the AR-Sanad 280K dataset, we address the narrator disambiguation in several experimental setups. The *hadith* narrator disambiguation is modeled as a multiclass classification problem with 18,298 class labels. We test different representations and models in our experiments. The best results were achieved by finetuning BERT-Based deep learning model (AraBERT). We obtained a 92.9 Micro F1 score and 30.2 *sanad* error rate (SER) on the validation set of our artificial *sanads* AR-Sanad 280K dataset. Furthermore, we extracted a real test set from the *sanads* of the famous six books in Islamic *hadith*. We evaluated the best model on the real test data, and we achieved 83.5 Micro F1 score and 60.6 *sanad* error rate.

**Keywords:** narrator disambiguation; Arabic dataset; natural language processing; deep learning; AraBERT; Islamic *hadith*



**Citation:** Mahmoud, S.; Saif, O.; Nabil, E.; Abdeen, M.; ElNainay, M.; Torki, M. AR-Sanad 280K: A Novel 280K Artificial Sanads Dataset for Hadith Narrator Disambiguation. *Information* **2022**, *13*, 55. <https://doi.org/10.3390/info13020055>

Academic Editor: Andreea Claudia Serban

Received: 23 November 2021

Accepted: 20 January 2022

Published: 23 January 2022

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



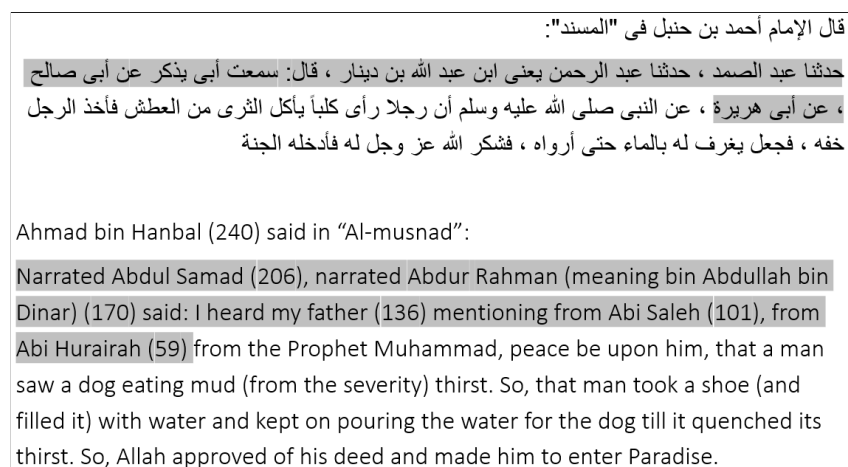
**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

In the Arabic language, the word “*hadith*” means speech or talk. In Islamic terminology, it refers to the words or deeds of Prophet Muhammad (PBUH) or his tacit approval or criticism of the actions of other people. The *matn* and the *sanad* are the primary components of all *hadiths*. The term *sanad* refers to the narration chain, a chronological list of the narrators who relayed the *hadith* from the era of Prophet Muhammad (PBUH) to the present day. The term *matn* refers to the main body of the *hadith*, the statement or report that needs to be delivered.

Figure 1 shows an example from “Musnad Imam Ahmad”, one of the nine most famous *hadith* books. Narrator Abi Hurairah is a companion of Prophet Muhammad who heard the statement directly from the Prophet. The first narrator, Abdul Samad, is a more recent narrator who died in 206. To illustrate how the *hadith* is transmitted through time by

the narrators, we added the death dates of each narrator in Hijri (the Islamic lunar calendar which begins from Prophet Muhammad migration to Madinah in 622 CE).



**Figure 1.** Example *hadith* from “Musnad Imam Ahmad” by Ahmad bin Hanbal. Top: *hadith* in Arabic. Bottom: translation of *hadith* in English. Every narrator’s death date is written in (.) using Hijri calendar.

*Hadiths* are of great importance in Islam. They are a major Islamic legislative source, second only to the Quran. Muslims rely on authentic *hadiths* to know what is allowed and what is prohibited. *hadiths* give a detailed explanation of how to practically apply Quranic principles since “Prophet Muhammad is seen as the ‘living Quran,’ the embodiment of God’s will in his behavior and words” [1].

Seeing that *hadiths* hold great influence, there was an urgent need to make sure that *hadiths* are not fabricated, to prevent people from being misguided by statements that are falsely attributed to Prophet Muhammad. To that end, the science of *hadith* emerged. *Hadith* science is a science that tackles the evaluation of *hadith* authenticity.

*Hadith* authentication relies heavily on the *sanad* and the reliability of the narrators. *Hadith* scholars believe that if the chain of narrators of a *hadith* fulfills five criteria, the *hadith* is to be accepted as authentic: continuity in the chain of narrators; integrity of character; infallible retention; freedom from any hidden defect; and safety from any aberrance [2,3]. The last two criteria apply to the *matn*, which is rarely examined by *hadith* scholars.

One of the challenges facing *hadith* scholars when dealing with the *sanad* is disambiguating the name of the narrator when only the first name is mentioned in the narration chain [4]. Identifying ambiguous names or “تعيين المهمل” is a branch of *hadith* science that deals with *unidentified narrators* (المهملون) in the *sanad*. This is different than dealing with unknown and *unnamed narrators* (example of *unidentified narrator*: “Said Muhammad” and example of *unnamed narrator*: “Said a man”). Narrators are considered *unidentified* or مهملون when someone refers to them by names that are similar to other narrators’, e.g., first name or surname.

An example is shown in Figure 2. There are two *sanads*; the name Hammad is mentioned in both of them. However, each one of them is referring to a different narrator, and they both lived in the same era. The first one is Hammad bin Salamah, and the second one is Hammad bin Zayd. In our narrators’ list, 112 other narrators share the first name Hammad. For more common names such as Muhammad or Ibrahim, the number goes even higher. This is a cause for concern. If two narrators are confused, this could lead to trusting someone who should not be trusted or vice versa.

<p>..., narrated Bahz, narrated <b>Hammad</b>, narrated Thabit, ...</p> <p style="text-align: center;"><u>sanad (1)</u></p> <p>..., narrated Abdullah bin Adbel-Wahhab said: narrated <b>Hammad</b>, from Ayyoub, ...</p> <p style="text-align: center;"><u>sanad (2)</u></p>
---

**Figure 2.** Illustration of ambiguous narrator name *Hammad* in two different *sanads*.

To identify a narrator, *hadith* scholars look at the narrators preceding and following the ambiguous name in the narration chain. It is also helpful to study the biographies of narrators. If they are still unable to identify the narrator, they could search for the same *matn* in different books to see if the narrator's name was spelled fully elsewhere.

After identifying the narrators, the authenticity of the *hadith* can be determined by examining the degree of reliability of each narrator in the *sanad*. Narrators' reliability can be determined by: (1) looking through their biographies, (2) looking at *Jarh* and *Taadeel* books to find what the scholars said about them, and (3) comparing what they had narrated with the *hadiths* of other narrators.

We introduce a new dataset that contains 279,625 artificial *sanads*. Artificial *sanads* are similar to real *sanads* but are created by making different combinations of narrators who narrated from each other. Each narrator in those *sanads* is tagged with an ID to disambiguate his name. More details on the creation process is in Section 3.1. The artificial *sanads* are created using narrators' data collected from *hadith* books and narrators' biographies that are made available in digital format at "عبد الله بن عبد العزيز للسنة النبوية المطهرة جامع خادم الحرمين الشريفين الملك"، the Custodian of the Two Holy Mosques for the Prophetic Sunnah. We refer to it as *Khadem Al-haramayn* website (<https://sunnah.alifta.gov.sa/>, accessed on 10 November 2021). This website was launched by الرئاسة العامة للبحوث العلمية والإفتاء, the General Presidency of Scholarly Research and Ifta.

### Contributions

Our main contributions in this paper are:

- Introducing a new Arabic dataset of artificial *sanads* (AR-Sanad 280K) with identified narrators. This dataset could be used to train systems to disambiguate narrators' names when their full names are not mentioned;
- Introducing a new dataset of real *sanads* that we use as a test set to evaluate models' performance on real data;
- We also present a systematic benchmark evaluation using AraBERT, a BERT-based model trained on a very large Arabic corpus. We also evaluate other models on the lite version of the AR-Sanad 280K dataset. This evaluation can be used by other studies to improve the models designed for the narrator disambiguation task.

According to *hadith* scholars, using a tool for disambiguation of narrators in *sanads* is very valuable for them, to semiautomate their efforts for studying different *hadith's sanads*. This saves a lot of time during *hadiths* investigation.

## 2. Related Work

Many works have been published in the field of *hadith* computation, using computational and machine learning methods to solve problems related to *hadith*. The survey by [4] covers major computational and NLP-based studies of *hadith*. In this section, we list research works that are most relevant to our work.

### 2.1. Hadith Computation

The work of [5] use information from the context (*sanad*), to find out if two narrators in different *sanads* with similar names are the same person. Comparing might help with building the narration graph. However, it is still hard to know the narrator's identities. For testing, they used *hadiths* from Sahih Al-Bukhari.

The work of [6] proposes a deterministic way to determine *hadith* authenticity using information from narrators' biographies. The authors classify sanads into three classes: *sahih*, *hasan*, and *dha'if* and used information from Taqrib al-Tahzeeb by Ibn Hajar to determine the reliability level of the narrators. To identify the narrators, they used data from [7]. To test their approach, they used 2180 *hadiths* from Sahih Al-Bukhari and 752 from Tirmizi. They achieved an accuracy of 99.6% on Sahih Al-Bukhari and 93.6% on Tirmizi.

Sahih Muslim and Sahih Al-Bukhari are the two most-trusted *hadith* books. All *hadiths* in Sahih Muslim and Sahih Al-Bukhari are authentic. Hence, in [8], the authors compiled a list of all their *sanads* and treated them as authentic. For any other *hadith*, if its *sanad* is in the list, they consider that *hadith* as authentic. If it is not on the list, the *hadith* is considered as inauthentic. However, this criterion leaves out other possible authentic *sanads*.

In the existing works in *hadith*, researchers used to build their own datasets collected from various *hadith* books. This makes it hard to compare the performance of different systems [9]. So recently, there have been some efforts to construct *hadith* datasets and make them available for research purposes.

In [10], the authors collected *hadiths* from four books in Arabic and English; Sahih Muslim, Sahih Al-Bukhari, Sunan Abi-Dawud, and Muwatta Imam Malik. They used regular expression to separate *matn* and *sanad*. In [11], the authors created a bilingual *hadith* corpus of 33,359 *hadiths* gathered from the six most famous *hadith* books. They used a segmentation tool to separate *matn* from *sanad* and included the degree of authenticity and other details about the *hadiths*.

## 2.2. Word Sense Disambiguation

The narrator disambiguation problem that we tackle in this paper has some similarity to Word Sense Disambiguation (WSD) where a word could have different meanings depending on the context. A name in the *sanad* could be referring to different people depending on other narrators in the *sanad*.

In [12], the authors explore two different strategies of integrating pretrained contextualized word representations for WSD: first, by using a straightforward way by finding a word in the training data with the closest contextual representation and assigning it the same sense, and second, passing the word representation through a linear classifier or filtering the representation vector through a gated linear unit. For the contextualized representation, they test two representations: the hidden vector of BERT in the last layer and a weighted sum of all hidden layers.

In [13], the authors create sense embeddings and use k-NN to determine the word sense. For each sense, its embedding is the concatenation of three vectors. The first embedding vector is determined by averaging contextual representation of words annotated with the given sense in the training data. The second embedding vector is determined by averaging contextual representation of words in the gloss of the given sense. The third vector is the fastText embedding of the given sense lemma.

In [14], the authors retrain BERT with the original masked token prediction jointly with masked word sense prediction.

In [15], the authors leverage relational information from lexical knowledge bases. They use the sum of the outputs of the last four layers of BERT as a vector representation of the target word. They feed this vector to a two-layer feedforward network. Instead of using the network output vector for prediction, they compute another vector such that the final score corresponding to a particular sense is a function of both its score and the scores of other senses related to it.

In [16], the authors use context sentence and gloss pairs to fine-tune BERT to determine whether the gloss belongs to the sense of the target word.

In [17], the authors train two encoders, context encoder and gloss encoder, both initialized with BERT. The context encoder produces a representation vector for each word in the context sentence. The gloss encoder produces sense representation. The target word

sense is the sense whose representation has the highest dot product score with the target word representation.

### 2.3. Arabic Named Entity Disambiguation

Named Entity Disambiguation (NED) or entity linking is a research area that focuses on linking mentions of entities in text to their corresponding objects in a knowledge base. Our problem is a specific subfield in this area where we focus only on narrator entities, and we use the data in *Khadem Al-haramayn* website as our knowledge base.

The work of [18] build AIDArabic, an extended version of AIDA [19] that is more suitable for performing NED on Arabic text.

In [20], the authors use information from DBpedia and Arabic Wikipedia. They search DBpedia using: Arabic entity mention, English Entity mention, or similar entities on Wikipedia. If none of this returned any entities, they search Wikipedia for articles about the entity. The information extracted from the article web page is used to populate an Arabic Ontology for future queries. If more than one entity is returned, they use other mentions of entities in the context to disambiguate the target entity.

In [21], the authors follow an approach that could be used with low-resource languages and applied it to the Arabic language. They used YAGO3 [22] as their knowledge base and extended the named-entity dictionary using JRC-Names [23] and Google Word-to-Concept repository [24]. They also trained a machine translation system to translate named entities. This helps to find entities that are mentioned in Arabic but only exist in the English knowledge base.

In these works, they try to link any mention of an entity, including a person name, to the corresponding object in a knowledge base. They deal with mentions of entities in regular textual data. Our problem is different since *sanads* contain only names. It is better to use domain-specific knowledge base.

None of these works address the problem of narrator disambiguation. In [25], the authors try to link narrator names with entities from DBpedia. However, they found it very lacking in this domain, a large number of narrators did not have any entities. In our AR-Sanad 280K dataset, we solve this problem by including 18,298 narrators from the list of narrators in *Khadem Al-haramayn* website.

*Khadem Al-haramayn* is a website that serves the *hadith* science. It has a great collection of *hadith* books in a digital format and offers a variety of services that help scholars with their research. One of the services they offer is narrator identification. Using rules known in *hadith* science, they identify all narrators in the *hadiths* they added to the website. In [26], the author explore the narrator identification service and how it could be used. They show the advantages of the service and propose ideas to improve it.

### 3. AR-Sanad 280K Dataset

In this section, we describe our new dataset and the creation process. Because the number of *sanads* in the six famous *hadith* books is only about 40K *sanads*, we did not use real *sanads* from *hadiths*. Instead, we created artificial *sanads* by listing chronologically the narrators who heard from each other. We used the artificial *sanads* for training and reserved *sanads* of real *hadiths* for testing.

The artificial *sanads* we use reflect real *sanads*. The names we use in the artificial *sanads* are the names of real narrators, and we do not refer to them by their nicknames or first names randomly. We refer to them by the names that were used in real *hadiths*. This means that some of those artificial *sanads* could be found in real *hadiths*. However, none of the *sanads* in the test set are included in the training data. In Section 4.3, we show how the models trained on our artificial AR-Sanad 280K dataset perform when dealing with real *sanads*.

### 3.1. Creating Artificial Sanads

First, we scraped narrators' data from the *Khadem Al-haramayn* website. The website has a list of 18,861 narrators and their information. Information about each narrator includes their full name, a list of the narrators they narrated from/to, their *appearance forms* (صور الورود), date of death, etc.

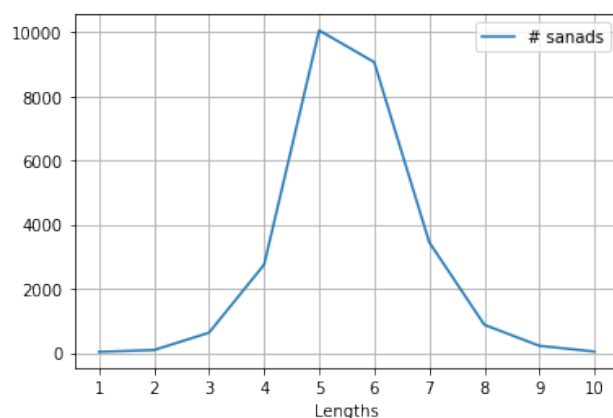
*Appearance forms* or صور الورود are short names or ways used to refer to a narrator when he is mentioned in a *sanad*. As an example, consider a narrator whose full name is: "Ibrahim bin Marwan bin Muhammad bin Hassan". However, he can be referred to by different *appearance forms* such as "Ibrahim bin Marwan", "Ibrahim bin Marwan Aldemashquey", "Ibrahim bin Marwan bin Muhammad Aldemashquey" or "Ibrahim bin Marwan bin Muhammad Altatarey". Other common *appearance forms* are أبيه meaning, his father, or جده meaning, his grandfather. When a narrator hears the *hadith* from his father or grandfather, he might refer to them using those forms without mentioning their names.

A few narrators from the list did not have sufficient information to include their names in any *sanad*. Thus, the final number of narrators that we used in the AR-Sanad 280K dataset is 18,298. In the end, every narrator has a unique ID that is used to form the artificial *sanads*.

To create the artificial *sanads* dataset, we go through the list of narrators one by one and do the following for each narrator ID:

1. We pick a random narrator ID that he/she narrated to and a random narrator ID that he/she narrated from;
2. For the two narrators IDs we picked, we select two other narrators IDs they narrated to/from;
3. For each narrator of the five narrators, we select a random name from their *appearance forms*;
4. The term [فاصل] (Means separator) is used as a separator between the five names, and the final *sanad* is in the form:  
name1 [فاصل] name2 [فاصل] ... [فاصل] name5
5. We repeat steps 1–4 a few times depending on the number of narrators they narrated to/form.

The above procedure produces artificial *sanads* of length five narrators. Similar procedures can be used to produce longer or shorter *sanad* chains. We created *sanads* of lengths (3–7) as those are the most common *sanad* lengths in real *hadiths*. Figure 3 shows the distribution of *sanad* lengths in ~30 k real *hadiths*.



**Figure 3.** Distribution of *sanad* lengths in a set of ~30 k real *sanads*.

Notice that we use a standard separator, [فاصل], and not the narration terms that usually appear in real *hadiths* such as the *hadith* in Figure 1. Some examples of narration terms are: narrated, said, mentioned, heard, etc. Extracting narrator names from *sanads*

is another topic discussed by many works [27–32], but it is not in the scope of our paper. Those narration terms do not help the identification process.

We show the steps to create the *sanads* in Figure 4. It also explains how we deal with boundary cases where the narrator is a companion of Prophet Muhammad who heard directly from him. In this case, there are no narrators that he narrated from. There is also the case where the narrators lived very recently and no other narrators heard from them, but the *hadiths* they narrated were written in books.

---

#### Algorithm 1 Creating length-5 sanad

---

**Input:** IDs, Names, Narrated to, Narrated from  
**Output:** Sanads

```

1: procedure CREATE SANAD(S)
2:   sanads ←
3:   for every id do
4:     if tos(id) and froms(id) not empty then
5:       to1 ← getRandom(tos(id))
6:       to2 ← getRandom(tos(to1))
7:       from1 ← getRandom(froms(id))
8:       from2 ← getRandom(froms(from1))
9:       sanads ← sanads + [to2, to1, id, from1, from2]
10:    else
11:     if tos(id) not empty then
12:       to1 ← getRandom(tos(id))
13:       to2 ← getRandom(tos(to1))
14:       to3 ← getRandom(tos(to2))
15:       to4 ← getRandom(tos(to3))
16:       sanads ← sanads + [to4, to3, to2, to1, id]
17:    else
18:     if froms(id) not empty then Do same as tos
return sanads

```

---

**Figure 4.** Steps to create sanads of length five.

### 3.2. Special Appearance Forms

Some *appearance forms* need special attention. When a narrator does not use names and calls to other narrators by their relation *lisuch aske* جدّه, أبيه which means his father and grandfather, respectively, we refer to this as using special *appearance forms*. If we used the same method as before to generate *sanads* with special forms, the special forms might be selected randomly when the narrator preceding them is not the son or the grandson. To avoid this, we make sure to select the right preceding narrator whenever we use a special *appearance form*.

The special *appearance forms* are frequently used in real *sanads*. In our dataset, we have 12,123 *sanads* with special forms.

### 3.3. Dataset Refinement

In this section, we explain the issues we faced while generating the data and how we fixed them.

- We removed duplicate *sanads*;
- We removed any name that was misspelled in *appearance forms*;
- After filtering the *appearance forms*, some narrators did not have *appearance forms*. We referred to them using their full names. If the name was too long we use only their first four names;
- We removed duplicate narrators who have identical information in the narrators' list, i.e., same full name, kunia, death date, etc.

### 3.4. Dataset Statistics

We summarize here some statistics of our dataset. To generate train and validation splits, we used scikit-learn (<https://scikit-learn.org/stable/>, accessed on 10 November 2021) implementation for multilabel stratification. We made sure that every narrator has at least one observation in the training data. The number of *sanads* in the train and validation set are 223,750 and 55,875 *sanads*, respectively.

We created overall 279,625 artificial *sanads* that included 18,298 narrators. In Table 1, we show narrators with the highest and lowest number of appearances in the data. There are 260 narrators who had only one observation in the data. The number of observations for each narrator depends on the number of narrators they narrated to/from (connections).

**Table 1.** Narrators with high or low observations in the AR-Sanad 280K dataset and the number of connections they have (i.e., number of narrators they narrated to/from).

Narrator	# Observations	# Connections
Sulayman bin Ahmad	12,968	1053
Omar bin Alkhatab	9231	899
Abu Muhammad Masud bin zayd	2	2
Daylam bin Abi Daylam	1	1

There are 167 narrators who have the name “Muhammad” in *appearance forms*. In Table 2, we show some of the most common names and the number of narrators who share them. There are 61,598 unique names in *appearance forms*. Only 3477 of them are shared by more than one narrator. The rest of them are unique to one narrator.

**Table 2.** Confusing *appearance forms* that are shared by many narrators. These numbers are obtained by analyzing data from the *Khadem Al-haramayn* website.

Appearance Forms	# Narrators
Muhammad	167
Abdullah	128
Ahmad	97
Abdurrahman	95
Ibrahim	94

### 3.5. Creating Lite Dataset

The size of the full dataset is about 280K *sanad*. This prevents us from evaluating multiple models due to the expected longer training time. Therefore, we constructed a lite version of our AR-Sanad 280K dataset. We built our lite version using the same algorithm in Figure 4 but restricted the number of narrators to 2222 narrators. Using the lite version, we were able to train different models and that led us to the final choice of using AraBERT over different alternatives.

## 4. Experiments

The most common method that is used by *hadith* scholars to identify narrators is by leveraging the information about the narrator’s students and teachers, i.e., the narrators he narrated to/from. There is a large number of possible combinations of narrators. It is hard to compile a list of all possible cases and how to deal with them. In these kinds of problems, it is preferable to use a machine learning model. It is much easier to let the model learn from data. In this section, we describe our experiments on the dataset using machine learning and deep learning methods.

### 4.1. Lite Dataset

In this section, we describe the experiments on the lite dataset in order to select the best performing models. We do three types of experiments.



#### 4.1.1. Static Embeddings

In the first part of the experiments, we compose a vector representation for each narrator and use it as the input to the classifier. We tested two classifiers, KNN with  $k = 3$  and Naive Bayes. We show the results in Table 3.

The vector representation we used is:

**FastText300:** For each narrator in the *sanad*, we average FastText embeddings of the tokens in his name.

**FastText600:** To encode some of the context information, we concatenate two vectors. The first one is the FastText300 representation of the narrator we want to classify. The second one is the FastText300 representation of the following or preceding narrator in the *sanad*.

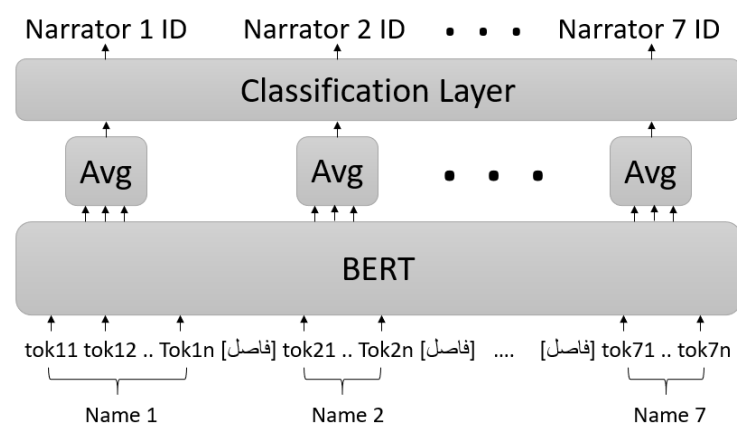
**Table 3.** Experiments on lite dataset using FastText.

Model	Accuracy
FastText600 + KNN	54.1
FastText600 + NB	65.5
FastText300 + KNN	<b>81.2</b>
FastText300 + NB	67

#### 4.1.2. AraBERT

Due to the prevalence of the BERT [33] model for text classification (and other NLP tasks), we evaluate AraBERT [34] on our AR-Sanad 280K dataset. AraBERT is a BERT-based model pretrained on a large-scale Arabic corpus. The size of vocabulary in AraBERT is 64k tokens. This makes a big difference in the performance since our data consists of names only. Notably, 61% of the tokens in our dataset are in Arabert's vocab. This is a good number considering that they are all names. AraBERT also does preprocessing using Farasa segmenter [35] to segment words into stems, prefixes and suffixes. In Section 4.3.1, we show the difference in performance when we do not do preprocessing with Farasa.

We use AraBERT to get a vector representing each narrator in the *sanad*. We do this by averaging tokens embeddings of the narrator name. Then, we feed this vector to a classification layer to determine the identity of the narrator. Figure 5 illustrates this process.



**Figure 5.** Classification process: At first every narrators' name is tokenized. Next, the BERT model produces token embeddings. Token embeddings that belong to the same narrator are averaged. Finally, a classification layer produces the output Narrator IDs. Better seen after zooming in.

We tried two settings for the model. First, we used AraBERT in the frozen setting. We used the embeddings to train a classification layer. In the second setting, we fine-tuned AraBERT and used the classification layer parameters from the previous experiment as the initial parameters.

Beside using neural networks for classification, we also tried KNN and Naive Bayes classifiers. We also tried building narrator embeddings (narrEmb) and used it together with

1-Nearest-Neighbor to determine narrators' identities. The results of these experiments are shown in Table 4 and show that tuning AraBERT with one classification layer gives the best performance.

**Table 4.** Experiments on lite dataset using AraBERT.

Model	Accuracy
AraBERT + KNN	85.5
AraBERT + NB	69.7
AraBERT + narrEmb + 1-NN	65.6
FrozenAraBERT + 1-layer NN	90.5
FrozenAraBERT + 2-layer NN	90.4
TunedAraBERT + 1-layer NN	<b>93.1</b>

#### 4.1.3. Other Deep Learning Models

Table 5 shows results of tuning other transformer models, namely AraElectra [36] and AIBERT [37]. Both models did not produce as good results as the AraBERT model since they both focus on optimizing model efficiency more than performance.

**Table 5.** Performance of different tuned transformer models.

Model	Accuracy
AraElectra [36]	92.1
AIBERT-Arabic [37]	92.5
AraBERT [34]	<b>93.1</b>

**AraElectra:** It uses *replaced token detection* as a pretraining task. This enables the model to learn from all input tokens instead of just the small masked-out subset. The objective of this work is to improve the efficiency of pretrained model without hurting the performance. The performance of a small-sized Electra model is not so far behind the BERT model.

**AIBERT:** It also focuses on model efficiency with reasonable performance. It reduces the number of BERT parameters by  $18\times$  and increased training speed  $1.7\times$ .

#### 4.2. Full Dataset

We report F1 scores on the validation set of the full AR-Sanad 280K dataset in Table 6 (Large) and top-k accuracy on Table 7. The fine-tuned AraBERT achieves 92.9 F1. We tested only AraBERT since it produces the best performance on the lite dataset. We observed that the fine-tuned AraBERT outperforms the frozen version. We also report the *sanad* error rate to evaluate the model ability to classify whole *sanads* correctly. *Sanad* Error Rate (SER) is the percentage of *sanads* that have at least one of their narrators misclassified. In Table 8, we show the percentage of *sanads* that have n narrators with wrong predictions.

**Table 6.** F1 scores and *sanad* error rate (SER) of the models trained on the large dataset. The *Large* section shows performance on the validation set of artificial *sanads*. The *Real* section shows performance on the test set of real *sanads*. FrozenAraBERT uses pretrained AraBERT with a classification layer. TunedAraBERT uses pretrained AraBERT and initialize the classification layer with the trained parameters from FrozenAraBERT and fine-tune both.

Dataset	Large			Real		
	MicroF1	MacroF1	SER	MicroF1	MacroF1	SER
Frozen	90.1	88.4	39.9	77.5	68	73.4
Tuned	<b>92.9</b>	<b>92.5</b>	<b>30.2</b>	<b>83.5</b>	<b>78.8</b>	<b>60.6</b>

**Table 7.** Evaluating TunedAraBERT: Top k accuracy.

<b>k</b>	<b>Val</b>	<b>Test</b>
1	92.9%	83.5%
3	97.4%	95.3%
5	98.4%	97.1%

**Table 8.** Evaluating TunedAraBERT: Percentage of *sanads* that have n narrators with wrong predictions.

<b># Wrong Predictions</b>	<b>Val</b>	<b>Test</b>
0 narrators	69.8%	39.4%
1 narrator	25.1%	36.8%
2 narrators	4.6%	17.6%
3 narrators	0.5%	5.2%
>3 narrators	0.1%	1%

#### Effect of Sanad Length

We take a look at how the *sanad* length affects model performance. We split the validation set by length and observed the change in performance. In Table 9, we report the results. We notice that the *sanad* error rate increases as the *sanad* becomes longer. This is expected since a larger number of narrators need to be identified correctly. We also notice that F1 scores become higher. The model becomes better at identifying individual narrators when it obtains more clues (narrators).

**Table 9.** Evaluating TunedAraBERT: F1 scores and SER on *sanads* of different lengths.

<b>Length</b>	<b>MicroF1</b>	<b>MacroF1</b>	<b>SER</b>
3	91.2	88.5	23.5
4	92.6	90.8	25.6
5	92.7	91.8	30.8
6	93.5	92.1	32.3
7	93.6	91.8	36.4

#### 4.3. Real Sanads Test Set

To evaluate the models ability to identify narrators in real *sanads*, we collected real *sanads* from the six most famous *hadith* books. We gather these data from *Khadem Al-haramayn* website. It encompasses a large collection of *hadith* books with identified narrators. Large groups of researchers and specialists in different fields worked together to develop this website [26]. The narrators' group had 30 members. They were responsible for work related to narrators including identifying *hadith* narrators, creating a database for them, etc., [38].

Table 10 shows the number of *sanads* we extracted from each book. After removing duplicate *sanads* and *sanads* with lengths less than 3 or more than 10, the total number of *sanads* in the test set is 27,056. We did not include *hadiths* that had more than one *sanad*. The list of 18,298 narrators is collected from a large number of *hadith* books. Only 6378 narrators were included in the test set, since the test set has only *sanads* from the six most famous *hadith* books. We report the results on the whole test set are in Table 6 and top-k accuracy on Table 7. In Table 11, more detailed results on each book are shown separately. In Table 8, we show the percentage of *sanads* that have n narrators with wrong predictions. We see that a large percentage of the *sanads* have only one misidentified narrator. This caused the SER to be as high as 60.6% when only 23.8% of the *sanads* more than one misidentified narrator.

**Table 10.** Number of *sanads* collected from the 6 most famous *hadith* books.

Book	# Sanads
Sahih Al-Bukhari	5674
Sahih Muslim	5189
Sunan Abi Dawud	4084
Al-Termizi	3588
Sunan Al-Nasai	4903
Sunan bin Majah	3756

**Table 11.** Evaluation of TunedAraBERT: Micro, Macro F1, sanad error rate, and Top-k accuracy on each book separately.

Book	MicroF1	MacroF1	SER	Top-1	Top-3	Top-7
Sahih Al-Bukhari	78.5	59.5	68.9	78.5	93.3	95.7
Sahih Muslim	84.3	71.9	60.2	84.3	96.6	98.1
Sunan Abi Dawud	84.9	79.1	57.6	84.9	95.3	97.1
Al-Termizi	88.3	80.7	48.7	88.3	97.4	98.5
Sunan Al-Nasai	80.2	72	70.5	80.2	93.2	95.7
Sunan bin Majah	87.9	82.8	50.5	87.9	97.2	98.4

#### 4.3.1. Effect of Farasa Segmenter

In this section, we observe the change in performance with and without preprocessing using the Farasa segmenter. First, we show examples of AraBERT tokenizer output in both cases in Figure 6.

<ul style="list-style-type: none"> <li>• Text: أبو القاسم سليمان بن أحمد الطبراني</li> <li>• + Farasa: 'أبو', 'ال+', 'قاسم', 'سليمان', 'بن', 'أحمد', 'ال+', 'طبراني'</li> <li>• - Farasa: 'أبو', 'الق', '##اسم', 'سليمان', 'بن', 'أحمد', 'ال+', '##بر', '##ني'</li> </ul>
<ul style="list-style-type: none"> <li>• Text: عبد الرحمن [فاصل] أمه</li> <li>• + Farasa: 'عبد', 'ال+', 'رحمن', 'فاصل', 'أم', 'ه+', 'فاصل', 'أمه'</li> <li>• - Farasa: 'عبد', 'الرح', '##من', ']', 'فاصل', ']', 'أمه'</li> </ul>
<ul style="list-style-type: none"> <li>• Text: Abu AlKasem Sulayman bin Ahmad AlTabarany</li> <li>• + Farasa: 'Abu', 'Al+', 'Kasem', 'Sulayman', 'bin', 'Ahmad', 'Al+', 'Tabarany'</li> <li>• - Farasa: 'Abu', 'Alk', '##asem', 'Sulayman', 'bin', 'Ahmad', 'AlTa', '##bara', '##ni'</li> </ul>
<ul style="list-style-type: none"> <li>• Text: Abd AlRahman [SEP] Ummih</li> <li>• + Farasa: 'Abd', 'Al+', 'Rahman', 'SEP', 'Ummi', '+h'</li> <li>• - Farasa: 'Abd', 'AlRah', '##man', 'SEP', 'Ummih'</li> </ul>

**Figure 6.** Examples of AraBERT tokenizer output. '+ Farasa': Farasa was used for preprocessing. '- Farasa': No preprocessing. Out-of-Vocab words are split, and the subwords are marked with ##. At the bottom are transliterations of the given examples.

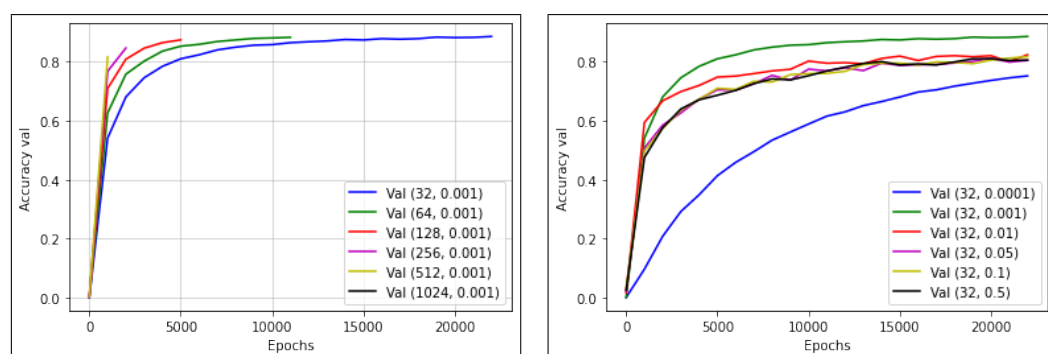
In the examples shown, Farasa separates ال (equivalent to *the* in the English language) and the possessive pronoun ه (equivalent to *his* in the English language) from the rest of the word. By separating any word from anything that is not an intrinsic part of it, Farasa helps improve the performance of models. In Table 12, we compare the performance of TunedAraBERT on the test set when we trained it with Farasa versus with no preprocessing.

**Table 12.** Evaluation of TunedAraBERT: Micro, Macro F1, and sanad error rate on the test set with/without Farasa segmenter.

Using Farasa	Micro F1	Macro F1	SER
Yes	83.5	78.8	60.6
No	81	74	66.8

#### 4.4. Implementation Details

For training the classification layer in FrozenAraBERT, we selected the best pair of batch size and learning rate from [32, 64, 128, 256, 512, 1024] and [ $1e-4$ ,  $1e-3$ , 0.01, 0.05, 0.1, 0.5]. We trained the model for five epochs for each pair of parameters and selected the pair that achieves the highest accuracy on the validation set. We show in Figure 7 the training curves of some of the parameters that we tested. When fine-tuning, we reduce the learning rate by a factor of 10. The parameters we used are shown in Table 13. The model was trained with Adam optimizer [39]. Due to restrictions on the RAM available, we used a batch of size 32 on the fine-tuned model when training on the full AR-Sanad 280K dataset.

**Figure 7.** Training with different hyperparameters. On the left are the training curves with different batch sizes and learning rate of 0.001. On the right, different learning rates and batch size of 32.**Table 13.** Hyperparameters used in training.

Model-Dataset	Batch Size	lr
Frozen-lite	32	$1e-3$
Tuned-lite	32	$1e-4$
Frozen-large	128	$1e-3$
Tuned-large	32	$1e-4$

For frozen and tuned AraBERT, we start with the selected learning rate. During training, we decrease the learning rate by a factor of two when the accuracy on the validation set is not improving. An initially large learning rate suppresses the network from memorizing noisy data while decaying the learning rate improves the learning of complex patterns [40]. Training stops after a certain number of trials decreasing the learning rate. Ten trials for frozen AraBERT and eight trials for fine-tuning.

## 5. Analysis

We explore some of the possible causes for error on the validation set. We examine the model TunedAraBERT performance on: narrators with low number of appearances in training data and narrators who have common or special *appearance forms*. We observed the following:

- First, we look at some of the predictions made by the model. In Figures 8 and 9, we show a few examples of false and true predictions and the narrators' true identities. We notice that in most cases the model's confidence level is higher for true

predictions than false ones. In total, 68.3% of all narrators were identified correctly with a confidence level of 90% or above. In total, 81.7% of the correct predictions have a confidence level of 90% or above. Only 12.9% of the false predictions have a confidence level of 90% or above;

- Special *appearance forms* could be a little confusing for models, since the narrator name is not stated. Only 71% of the narrators that appeared in a special form were classified correctly;
- Figure 10 shows examples of narrators that were not identified correctly, but their true identities were in the top five most probable ones. Most of them are called by common short names;
- As we shown in Table 2, there are many narrators who have similar *appearance forms*. There were 3669 instances of the names showed in the table; only 26% were correctly identified. We hope that our AR-Sanad 280K dataset could be used to build better systems that can manage to avoid such errors.

Sanad	True identity	Prediction	Confidence
محمد بن عبد الرحمن بن ثوبان الأنصاري [فاصل] رفاعة [فاصل] أبو سعيد بن مالك الخدري	أبو مطيع بن عوف ، قيل اسمه : رفاعة ، وقيل فلان بن رفاعة	رفاعة بن رافع بن مالك بن العجلان بن عمرو بن عامر بن زريق	0.9855
سليمان بن الأشعث السجستاني [فاصل] محمد مسكين [فاصل] محمد بن أبي رجاء العباداني [فاصل] زياد	زياد بن كليب	زياد بن عبد الله بن الطفيل	0.4659
الحسن [فاصل] أبو عبد الرحمن أحمد بن شعيب الشيباني [فاصل] محمد بن ميمون الخياط المكي [فاصل] سفيان بن سعيد بن مسروق [فاصل] الحارث	الحارث بن يزيد	الحارث بن عبد الله ، وقال بعضهم : الحارث بن عبيد	0.3523
أبو عمر عبد الوهاب بن القاسم بن محمد بن مهرة [فاصل] سليمان بن أحمد [فاصل] علي [فاصل] إبراهيم بن عبد الله بن حاتم [فاصل] محمد بن دينار أبو بكر العلاني [فاصل] أبو إسماعيل العبيدي	علي بن عبد العزيز بن المرزبان بن سابور	علي بن سعيد بن بشير بن مهران	0.5654
سفيان بن سعيد بن مسروق الثوري [فاصل] بريد بن عبد الله [فاصل] جده [فاصل] البراء بن عازب الأنصاري	أبو بردة بن أبي موسى ، قيل اسمه : عامر بن عبد الله بن قيس ، وقيل : الحارث	عبد الله بن عمر بن الخطاب	0.6165

Sanad	True identity	Prediction	Confidence
Muhammad bin Abd Al-Rahman bin Thawban Al Ansary [SEP] Rifaa [SEP] Abu Said bin Malik AlKhudary	Abu Mutei bin Auf, said to be named: Rifaa, Fulan bin Rifaa	Rifaa bin Rafei bin Malik bin Al Ajlan bin Amr bin Aamer bin Zurayk	0.9855
Sulayman bin Al Ashaath Al Sajstany [SEP] Muhammad Miskin [SEP] Muhammad bin Abi Rajaa Al Abadany [SEP] Zeyad	Zeyad bin Kulayb	Zeyad bin Abdillaha bin Al Tufayl	0.4659
Al Hasan [SEP] Abu Abd Alrahman Ahmad bin Shuaib Al Shaybany [SEP] Muhammad bin Maymun Al Khayyat Al Makki [SEP] Sufyan bin Said bin Masruq [SEP] Al Harith	Al Harith bin Yazeed	Al Harith bin Abdillaha, said to be Al Harith bin Ubayd	0.3523
Abu Amr Abd Al Wahhab bin Al Kasem bin Muhammad bin Mahra [SEP] Sulayman bin Ahmad [SEP] Ali [SEP] Ibrahim bin Abdillaha bin Hatem [SEP] Muhammad bin Dinar Abu Bakr Al Alaei [SEP] Abu Ismail Al Abdy	Ali bin Abd AlAziz bin Al Marzaban bin Sabur	Ali bin Said bin Basheer bin Mahran	0.5654
Sufyan bin Said bin Masruq Al Thawry [SEP] Burayd bin Abdillaha [SEP] His grandfather [SEP] Al Baraa bin Azeb Al Ansary	Abu Burda bin Abi Musa, said to be named: Amer bin Abdillaha bin Kays, Al Harith	Abdullah bin Umar bin Al Khattab	0.6165

Figure 8. Examples of false predictions for narrators marked in red, their true identity, the false identity predicted by the model and the model’s confidence in the prediction.

Sanad	Confidence
أبو أحمد الدمشقي [فاصل] أبو الزبير [فاصل] ابن كعب Abu Ahmad Al-Demashqy [SEP] Abu Al-zubayr [SEP] bin Kaab	0.9984
أبو داود [فاصل] محمد بن إسحاق بن محمد المسيبي [فاصل] أبي Abu Dawud [SEP] Muhammad bin Ishac bin Muhammad Al-Masiby [SEP] Abi	0.8487
إبراهيم بن إسحاق [فاصل] روح [فاصل] زهير بن محمد التميمي [فاصل] نعيم بن عبد الله المجمر [فاصل] سعيد بن المسيب Ibrahim bin Ishaq [SEP] Raoh [SEP] Zuhayr bin Muhammad Al-Tamimy [SEP] Nuaim bin Abdillah [SEP] Said bin Al-Mysayyeb	0.9999
الشافعي [فاصل] ابن أبي يحيى يعني إبراهيم [فاصل] عبيد الله بن تمام [فاصل] زينب بنت نبيط بن جابر امرأة أنس بن مالك [فاصل] فارعة بنت أبي أمامة Al-Shafei [SEP] bin Abi Yahya meaning Ibrahim [SEP] Obaydillah bin Tamam [SEP] Zaynab bint Nubayt bin Jabir Imaraat Anas bin Malik [SEP] Faria bint Abi Umamah	1.0000
حسان بن نوح [فاصل] عبد الله بن بسر الخزاعي [فاصل] أبيه [فاصل] الصماء Hassan bin Nuh [SEP] Abdullah bin bisr AlKhuzai [SEP] Abih [SEP] AlSammaa	0.9962

**Figure 9.** Examples of true predictions for narrators marked in green and the model's confidence in the prediction.

<ul style="list-style-type: none"> <li>• يحيى بن بكير [فاصل] الليث [فاصل] عقيل [فاصل] ابن شهاب [فاصل] عروة بن الزبير [فاصل] عائشة</li> <li>• عمرو بن خالد [فاصل] الليث [فاصل] يزيد [فاصل] أبي الخير [فاصل] عبد الله بن عمرو</li> <li>• خالد بن مخلد [فاصل] سليمان [فاصل] عبد الله بن دينار [فاصل] ابن عمر</li> <li>• أبو نعيم الفضل بن دكين [فاصل] شيبان [فاصل] يحيى [فاصل] أبي سلمة [فاصل] أبي هريرة</li> </ul>
<ul style="list-style-type: none"> <li>• Yahya bin Bakir [SEP] Al Layth [SEP] Ukayl [SEP] bin Shihab [SEP] Urwa bin Al Zubayr [SEP] Aisha</li> <li>• Amr bin Khaled [SEP] Al Layth [SEP] Yazeed [SEP] Abi Al Khayr [SEP] Abdullah bin Amr</li> <li>• Khaled bin Makhlad [SEP] Sulayman [SEP] Abdullah bin Dinar [SEP] bin Omar</li> <li>• Abu Nuaim Al Fadl bin Dukayn [SEP] Shayban [SEP] Yahya [SEP] Abi Salama [SEP] Abi Hurayra</li> </ul>

**Figure 10.** Examples of narrators whose true identities were not on top but were still in the five most probable identities.

## 6. Conclusions

We presented a new dataset of artificial *sanads* that could help models identify narrators with ambiguous names. Narrator disambiguation is very important in order to authenticate *hadiths*. This dataset could be used to build systems that could help *hadith* scholars who are working on this problem.

We fine-tuned AraBERT on our dataset and used it to classify narrators with 92.9 micro F1 and 92.5 macro F1 on the validation set. We also created a test set of real *sanads* from the six most famous *hadiths* books and tagged the narrators with their IDs. Our best model achieved micro and macro F1 scores of 83.5 and 78.8, respectively, on the test set.

**Author Contributions:** Conceptualization, O.S., E.N., M.A., M.E. and M.T.; data curation, S.M. and M.T.; formal analysis, S.M.; investigation, O.S., E.N., M.A., M.E. and M.T.; methodology, S.M., O.S., E.N., M.A., M.E. and M.T.; project administration, E.N. and M.E.; software, S.M.; supervision, O.S., E.N., M.A., M.E. and M.T.; validation, S.M., M.T.; visualization, S.M. and M.T.; writing—original draft, S.M. and M.T.; writing—review and editing, S.M., O.S., E.N. and M.T. All authors have read and agreed to the published version of the manuscript.

**Funding:** The authors extend their appreciation to the Deputyship for Research and Innovation, Ministry of Education in Saudi Arabia for funding this research work through the project number 20/18.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Data is available at <https://github.com/somaia02/Narrator-Disambiguation>.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Esposito, J.L. *The Future of Islam*; Oxford University Press: Oxford, UK, 2010.
2. Khan, I.A. *Authentication of Hadith: Redefining the Criteria*; Iiit: Herndon, VA, USA, 2010.
3. مقدمة النووي في علوم الحديث: وهي مقدمة على صحيح مسلم . 1996 .
4. Azmi, A.M.; Al-Qabbany, A.O.; Hussain, A. Computational and natural language processing based studies of hadith literature: A survey. *Artif. Intell. Rev.* **2019**, *52*, 1369–1414. [CrossRef]
5. Astari, A.; Bijaksana, M.A.; Suryani, A.A. Analysis Name Entity Disambiguation Using Mining Evidence Method. *Paradig. J. Inform. Komput.* **2020**, *22*, 101–108. [CrossRef]
6. Azmi, A.M.; AlOfaidly, A.M. A novel method to automatically pass hukm on hadith. In Proceedings of the 5th International Conference on Arabic Language Processing (CITALA'14), Oujda, Morocco, 26–27 November 2014; pp. 26–27.
7. Al-Azami, M.M. A note on work in progress on computerization of hadith. *J. Islam. Stud.* **1991**, *2*, 86–91. [CrossRef]
8. Alias, N.; Abd Rahman, N.; Nor, Z.; Alias, M. Searching algorithm of authentic chain of narrators' in Shahih Bukhari book. In Proceedings of the International Conference on Applied Computing, Mathematical Sciences and Engineering (ACME 2016), Johor Bahru, Malaysia, 30–31 May 2016.
9. Luthfi, E.T.; Suryana, N.; Basari, A.H. Digital hadith authentication: A literature review and analysis. *J. Theor. Appl. Inf. Technol.* **2018**, *96*, 5054–5068.
10. Mahmood, A.; Khan, H.U.; Alarfaj, F.K.; Ramzan, M.; Ilyas, M. A multilingual datasets repository of the hadith content. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 165–172. [CrossRef]
11. Altammami, S.; Atwell, E.; Alsalka, A. Constructing a Bilingual Hadith Corpus Using a Segmentation Tool. In Proceedings of the 12th Language Resources and Evaluation Conference, Marseille, France, 20–25 June 2022; pp. 3390–3398.
12. Hadiwinoto, C.; Ng, H.T.; Gan, W.C. Improved Word Sense Disambiguation Using Pre-Trained Contextualized Word Representations. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 5297–5306.
13. Loureiro, D.; Jorge, A. Language Modelling Makes Sense: Propagating Representations through WordNet for Full-Coverage Word Sense Disambiguation. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 5682–5691.
14. Levine, Y.; Lenz, B.; Dagan, O.; Ram, O.; Padnos, D.; Sharir, O.; Shalev-Shwartz, S.; Shashua, A.; Shoham, Y. SenseBERT: Driving Some Sense into BERT. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Virtual Online, 5–10 July 2020; pp. 4656–4667.
15. Bevilacqua, M.; Navigli, R. Breaking through the 80% glass ceiling: Raising the state of the art in word sense disambiguation by incorporating knowledge graph information. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Virtual Online, 5–10 July 2020; pp. 2854–2864.
16. Huang, L.; Sun, C.; Qiu, X.; Huang, X.J. GlossBERT: BERT for Word Sense Disambiguation with Gloss Knowledge. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 3509–3514.
17. Blevins, T.; Zettlemoyer, L. Moving Down the Long Tail of Word Sense Disambiguation with Gloss Informed Bi-encoders. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Virtual Online, 5–10 July 2020; pp. 1006–1017.
18. Yosef, M.A.; Spaniol, M.; Weikum, G. AIDArabic A Named-Entity Disambiguation Framework for Arabic Text. In Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP), Doha, Qatar, 25 October 2014; pp. 187–195.
19. Hoffart, J.; Yosef, M.A.; Bordino, I.; Fürstenau, H.; Pinkal, M.; Spaniol, M.; Taneva, B.; Thater, S.; Weikum, G. Robust disambiguation of named entities in text. In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, Edinburgh, UK, 27–31 July 2011; pp. 782–792.
20. Al-Smadi, M.; Talafha, B.; Qawasmeh, O.; Alandoli, M.N.; Hussien, W.A.; Guetl, C. A hybrid approach for Arabic named entity disambiguation. In Proceedings of the 15th International Conference on Knowledge Technologies and Data-Driven Business, Graz, Austria, 21–22 October 2015; pp. 1–4.
21. Gad-Elrab, M.H.; Yosef, M.A.; Weikum, G. Named entity disambiguation for resource-poor languages. In Proceedings of the Eighth Workshop on Exploiting Semantic Annotations in Information Retrieval, Melbourne, Australia, 23 October 2015; pp. 29–34.



22. Mahdisoltani, F.; Biega, J.; Suchanek, F.M. A Knowledge Base from Multilingual Wikipedias–Yago3. Technical Report, Technical Report, Telecom ParisTech. Available online: <https://suchanek.name/work/publications/cidr2015.pdf> (accessed on 15 November 2021).
23. Steinberger, R.; Pouliquen, B.; Kabadjov, M.; Belyaeva, J.; van der Goot, E. JRC-NAMES: A Freely Available, Highly Multilingual Named Entity Resource. In Proceedings of the International Conference Recent Advances in Natural Language Processing, Hissar, Bulgaria, 12–14 September 2011; pp. 104–110.
24. Spitkovsky, V.I.; Chang, A.X. A cross-lingual dictionary for english wikipedia concepts. In Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12), Istanbul, Turkey, 21–27 May 2012.
25. Prasetio, A.; Bijaksana, M.A.; Suryani, A.A. Name Disambiguation Analysis Using the Word Sense Disambiguation Method in Hadith. *Edumatic J. Pendidik. Inform.* **2020**, *4*, 68–74. [\[CrossRef\]](#)
26. Bin Ibrahim Saif, O. The Attention Given to Al-Muhmaluun (the Unspecified) Narrators in the Program of the Custodian of the Two Holy Mosques for the Prophetic Sunnah. *Islam. Univ. J.* **2020**, *1*, 379–429.
27. Shukur, Z.; Fabil, N.; Salim, J.; Noah, S.A. Visualization of the hadith chain of narrators. In *Proceedings of the International Visual Informatics Conference*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 340–347.
28. Boella, M.; Romani, F.R.; Al-Raies, A.; Solimando, C.; Lancioni, G. The SALAH Project: Segmentation and linguistic analysis of Hadith Arabic texts. In *Proceedings of the Asia Information Retrieval Symposium*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 538–549.
29. Siddiqui, M.A.; Saleh, M.; Bagais, A.A. Extraction and visualization of the chain of narrators from hadiths using named entity recognition and classification. *Int. J. Comput. Linguist. Res* **2014**, *5*, 14–25.
30. Alhawarat, M. A domain-based approach to extract Arabic person names using n-grams and simple rules. *Asian J. Inf. Technol.* **2015**, *14*, 287–293.
31. Hamam, H.; Othman, M.T.B.; Kilani, A.; Ben, M. Data mining in Sciences of the prophet’s tradition in general and in impeachment and amendment in particular. *Int. J. Islam. Appl. Comput. Sci. Technol.* **2015**, *3*, 9–16.
32. Najeeb, M.M. Multi-agent system for hadith processing. *Int. J. Softw. Eng. Appl.* **2015**, *9*, 153–166. [\[CrossRef\]](#)
33. Kenton, J.D.M.W.C.; Toutanova, L.K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the NAACL-HLT, Minnesota, MN, USA, 2–7 June 2019; pp. 4171–4186.
34. Antoun, W.; Baly, F.; Hajj, H. AraBERT: Transformer-based Model for Arabic Language Understanding. In Proceedings of the LREC 2020 Workshop Language Resources and Evaluation Conference, Marseille, France, 11–16 May 2020; p. 9.
35. Abdelali, A.; Darwish, K.; Durrani, N.; Mubarak, H. Farasa: A fast and furious segmenter for arabic. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, San Diego, CA, USA, 12–16 June 2016; pp. 11–16.
36. Antoun, W.; Baly, F.; Hajj, H. AraELECTRA: Pre-Training Text Discriminators for Arabic Language Understanding. In Proceedings of the Sixth Arabic Natural Language Processing Workshop, Kyiv, Ukraine, 19 April 2021; pp. 191–195.
37. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. Albert: A lite bert for self-supervised learning of language representations. *arXiv* **1909**, arXiv:1909.11942.
38. المهمات والمناهج والضوابط العملية : مجموعات العمل . 2014 .
39. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
40. You, K.; Long, M.; Wang, J.; Jordan, M.I. How does learning rate decay help modern neural networks? *arXiv* **2019**, arXiv:1908.01878.