MDACE: MIMIC Documents Annotated with Code Evidence

Anonymous ACL submission

Abstract

The accuracy of Computer-Assisted Coding (CAC) systems has improved significantly in recent years, thanks to advances in machine learning technologies. Yet simply predicting a set of final codes for a patient encounter is insufficient as CAC systems are required to provide supporting textual evidence to justify the billing codes. A model able to produce accurate and reliable supporting evidence for each code would be a tremendous benefit. However, a human annotated code evidence corpus is extremely difficult to create because it requires specialized knowledge. In this paper, we introduce MDACE, the first publicly available code evidence dataset, which is built on a subset of the MIMIC-III clinical records. The dataset – annotated by professional medical coders – consists of 302 Inpatient charts with 3,934 evidence spans and 52 Profee charts with 5,563 evidence spans. We implemented several evidence extraction methods based on the EffectiveCAN model (Liu et al., 2021) to establish baseline performance on this dataset. MDACE can be used to evaluate code evidence extraction methods for CAC systems, as well as the accuracy and interpretability of deep learning models for multi-label classification. We believe that the release of MDACE will greatly improve the understanding and application of deep learning technologies for medical coding and document classification.

1 Introduction

Computer-Assisted Coding (CAC) uses Natural Language Processing (NLP) techniques to extract procedure and diagnosis codes from the documentations of patient encounters. MIMIC-III (Medical Information Mart for Intensive Care) (Johnson et al., 2016) is an open-access dataset comprised of hospital records associated with patients admitted to the critical care units of the Beth Israel Deaconess Medical Center. For each patient record/chart, the data related to billing includes diagnostic codes, procedure codes, clinical notes by care providers (discharge summaries, radiology and cardiology reports, nursing notes, etc.), and other patient demographic data. The MIMIC records were originally coded with the numerical-based code system ICD-9 (International Classification of Diseases), which contains approximately 14,000 codes overall.

Since the release of MIMIC-III, there has been a surge of research on using machine learning (ML) models to predict the billing codes based on the clinical text (Ji et al., 2022). However, the MIMIC database does not contain the association between the billing codes and the clinical notes, i.e., the specific narratives in the notes supporting the codes are not present. CAC systems are required to extract text evidence to support the generated billing codes. There is no dataset for reference code evidence as it requires medical coding expertise and is costly to build. As a result, work until this point can only illustrate qualitatively that their models can extract text evidence that look reasonable to humans. This approach is time-consuming and makes the comparison of different methods extremely difficult. The need for a reference evidence dataset is obvious.

In many parts of the world, the ICD-9 code system is out of date. Most countries are currently using the much more robust alphanumeric code system, ICD-10. The U.S. version, ICD-10-CM, has approximately 69,000 codes while the procedures (PCS) have about 82,000 codes. This dramatic increase in number can be attributed to the addition of modifiers for disorders such as laterality, severity, acuity, and sequence for injuries. While the principles of coding remained the same, the transition from ICD-9-CM to ICD-10-CM between 2014 and 2015 changed the way medical coders read documentation and code from them. While the entire chart should be read to understand the patient's story, only documents generated as a result of a face-to-face visit with an allowable provider should be reviewed for direct ICD-10 code abstraction. This includes Progress Notes, History and Physicals, Consults and Operatives Notes, etc. For procedure code selection, only a procedure or operative note is acceptable.

For these reasons, the ML models trained on the MIMIC-III discharge summaries to predict ICD-9 codes have little value for medical coding in reality. MIMIC-IV (Johnson et al., 2020) improved upon MIMIC-III in many ways, one of which is the addition of ICD-10 codes. But the clinical notes associated with the patient records have yet to be released.

In this paper, we introduce MDACE, the first publicly available code evidence dataset[1] built on a subset of the MIMIC-III clinical records. The dataset contains evidence spans for diagnosis and procedure codes annotated by professional medical coders. Each span contains the billing code and the text offsets in the respective clinical note. We provide Python scripts for merging our evidence representation with the MIMIC NOTEEVENTS table to obtain the true evidence so as to comply with *The PhysioNet Credentialed Health Data License*. To broaden its use, we automatically map between ICD-10 and ICD-9 codes with evidence so that the evidence can potentially be used with the MIMIC-IV corpus. MDACE addresses a critical need for CAC research to be able to automatically evaluate the code evidence generated by ML models.

2 Related Work

With the recent increased attention to the interpretability of deep learning models, datasets containing explanations in different forms (highlights, free-text, structured) have been curated. Wiegreffe and Marasovic (2021) provide a list of 65 datasets for various explainable NLP tasks, and Feldhus et al. (2021) present the results of different explanation generation models trained on these datasets. However, none of these datasets covers evidence for medical coding.

Many works have used private datasets for the development of evidence generation methods for medical coding, e.g., Sen et al. (2021). However, these datasets are not publicly available, and can't be used to improve the research on evidence extraction. Searle et al. (2020) used a semi-supervised approach to create a silver-standard dataset of clinical codes, from only the discharge diagnosis sections of the MIMIC-III discharge summary notes, with a small sample validated by humans.

There has been a surge in neural network models for automatic medical coding in the past several years. Mullenbach et al. (2018) first introduced a convolutional neural net with an attention mechanism, where the code (label) dependent attention weights were used as token importance measure for the model interpretability. Liu et al. (2021) extended on this work by incorporating the squeeze-and-excitation network (Hu et al., 2018) into the text encoder to obtain better contextual text representations. Xie et al. (2019) used the multi-scale convolutional attention while Vu et al. (2020) proposed to combine Bi-LSTM and an extension of structured self-attention mechanism for ICD code prediction. Some other recent models that achieved the state-of-the-art results on the MIMIC-III full code set include Kim and Ganapathi (2021); Hu et al. (2021); Yuan et al. (2022). There are also a large number of Transformer based models for medical coding, e.g., (Liu et al., 2022; Pascual et al., 2021), but they often only predict the top 50 codes and therefore have little value to solving real-world CAC problems. One exception is PLM-ICD (Huang et al., 2022), which uses domain-specific pretraining, segment pooling and label-aware attention to tackle the challenges of coding and improve performance. However, this model cannot extract phrase level evidence for the ICD codes.

Many of the above works use the attention weights to identify the text snippets that justify code predictions. But there is no quantitative evaluation of the quality of the snippets.

Works that use semi-supervised learning for explanation tasks in NLP include (Zhong et al., 2019; Pruthi et al., 2020; Segal et al., 2020), where Segal et al. (2020) use a linear tagging model for identifying answer snippets in question answering. Although they are not directly related to medical coding, we can apply their approaches for evidence extraction with the help of the MDACE dataset.

3 Challenges and Solutions

MIMIC-III poses a number of challenges for creating a reference code evidence dataset. In this section, we discuss these challenges including the coding specialties and code systems, and describe our solutions and process to increase the usability of MDACE.

[1] Link to the dataset will be provided in the final submission.

3.1 Coding Specialties

MIMIC-III contains both ICD-9 codes which are used for inpatient coding, and CPT (Current Procedure Terminology) codes, which are maintained by the American Medical Association (AMA) and used for outpatient facility and professional fee (Profee) billing in the U.S. There are approximately ten thousand CPT-4 codes. It was necessary to have different coders for each of these tasks (Inpatient vs. Profee) because it is unusual that one person be experienced in both areas. This means that inpatient coders tend to be more skilled ICD coders, while profee coders are often skilled CPT coders within their domain. ICD codes are also applied to profee charts to meet medical necessity requirements which ensure that the patient's bill is paid by insurance companies.

For this reason, we hired two coding teams with two professional coders each for Inpatient and Profee coding respectively. Although both teams coded diagnosis codes, the actual codes can be different due to different coding rules.

For either coding scenario, a coder usually looks for sufficient evidence that supports a code and ignores equally good evidence that she comes across later to save the time spent on each chart. This poses a challenge for evaluating CAC systems which can generate multiple pieces of evidence for a code that may or may not overlap with the *sufficient* reference evidence. To overcome this challenge but still finish the annotations in a reasonable time frame, we asked our coders to annotate sufficient evidence for Inpatient coding but complete evidence for Profee coding.

3.2 Code Mappings

We explained in Section 1 that MIMIC-III was coded in ICD-9, which has been discontinued. Updating the MIMIC-III dataset with ICD-10 codes and evidence will benefit research that targets real-world coding problems. MDACE is designed to contain evidence for both ICD-9 and ICD-10 codes so that it can be used to evaluate evidence extraction of CAC models trained on MIMIC-III, and also models that can predict ICD-10 codes, e.g., trained on MIMIC-IV once the notes are released.

We chose to use ICD-10 for annotation because firstly, most coders are more familiar with the ICD-10 code system, and secondly, ICD-10 codes are more specific, so the mapping from an ICD-10 code to ICD-9 would be less ambiguous than the other way round. Our coders annotated a subset of the MIMIC-III charts with ICD-10 codes and their evidence, which were then automatically mapped to ICD-9 through the General Equivalence Mappings (GEMs)[2] (Center for Medicare & Medicaid Services, 2009). GEMs contain six types of mappings, including Identical match, Approximate match, Combination map, and No Map, etc. To ensure the quality of code mapping, we follow this process to backward map ICD-10 to ICD-9:

1. Use the identical match or single approximate match from an ICD-10 to ICD-9 code;
2. When more than one mapping exists, choose the ICD-9 code that is in the MIMIC-III code set. If none of the mapped codes is in MIMIC, choose the code with the description that overlaps the most with that of the ICD-10 code;
3. When no mapping exists, use the mapped ICD-9 code of the parent ICD-10 code.

This process allows all annotated ICD-10 codes to be mapped except for two in our dataset.

3.3 Annotation Workflow

It is well known that medical coding is an extremely complex task, and there is often disagreement among coders. Given the large number of notes and codes in each MIMIC-III record (Su et al., 2019), it is impractical for our coders to first decide the best ICD-10 code for a MIMIC ICD-9 code and then annotate the narrative evidence in clinical notes for that code. Therefore, our coders followed their natural workflow of coding each chart from scratch to save time. However, the original MIMIC codes and their possible ICD-10 mappings were made available to them. After completing a chart, if there were MIMIC codes that were not accounted for, they could reference those left-overs, re-review the chart for evidence and annotate accordingly. If the coders could not find evidence after reviewing again, for example, the required note was missing, they simply made a note in their coding reports.

We used a tool called INCEpTION[3] to help our coders to review and annotate MIMIC charts. This tool allows them to browse through the clinical notes, highlight text spans and assign labels (billing

[2] GEMs are a comprehensive translation dictionary developed by multiple health organizations in the U.S. to effectively translate between the ICD-9 and ICD-10 codes.

[3] https://inception-project.github.io

codes) to the spans. The annotation guideline is illustrated in Appendix B.

We sampled a subset of the full code test set of Mullenbach et al. (2018) so as to build a dataset for evaluating code evidence. Depending on the size of the resulting dataset, it can also be used for training extraction methods. We randomly sampled batches of 50 charts from the test set, and extracted all clinical notes eligible for coding for each chart rather than just the discharge summaries. Our coders worked on one batch at a time, and the project lasted two months.

3.4 Inter-Annotator Agreement

As the first step of the annotation process, we measured the inter-annotator agreement to assess the reliability of the annotations. To quantify the quality of annotations, two coders independently annotated sufficient (for Inpatient) or complete (for Profee) evidence for the same three charts, and we measured their agreement. Next, they reviewed each other's annotations on where they disagreed to investigate the reasons for disagreement and see if they could reach an agreement. If they still disagreed, their supervisor made the final call. Once all disagreements were resolved, the coders started working on the first batch of charts following the same coding practice.

We used Krippendorf's α (Krippendorff, 2004) as an agreement measure, as it allows for assigning multiple labels to a span, which could be the case in medical coding. The punctuation marks in annotations were discarded in the calculation. The agreement for initial and final coding are given in Table 1, where the α values higher than 0.80 could be interpreted as strong agreement.

We observed two sources that accounted for the low initial agreement. One source is that the coders annotated the same or similar evidence from different locations of the same documents or in different documents of the same chart. For example, two coders annotated G60.8 for "idiopathic generalized neuropathy", one from the Physician Initial Consult Note, while the other from the Physician Surgical Admission Note. Both notes are valid for coding. Another example is that one coder assigned I46.9 for "Asystole" documented in the Discharge Summary while the other assigned the same code for "cardiac arrest" from the Physician Initial Consult Note. Both diagnosis terms are correct for I46.9. These cases were resolved in the re-review process, and should be treated as agreements.

	Inpatient	Profee
Encounter #1	0.63	0.34
Encounter #2	0.90	0.23
Encounter #3	0.18	0.07
All	0.51	0.24
After Review	0.97	0.96

Table 1: Krippendorf's α for inter-annotator agreement measures

Annotated	Inpatient	Profee
Encounters	302	52
Documents	604	588
ICD-9 Codes	918	652
ICD-10 Codes	1,024	734
Evidence for ICD-9	3,934	5,563
Evidence for ICD-10	3,936	5,563

Table 2: Summary of MDACE

The other source of disagreement came from external cause codes and symptom codes, which are not essential for billing, so some coders chose to code them while others did not.

For Profee coding, the initial disagreement was also due to the lack of experience of one coder. An example is that one coder assigned the code S04.40XA for "traumatic 6th nerve palsy" documented in the Discharge Summary whereas the other assigned the code H49.20 for the same diagnosis which is incorrect. The disagreement was resolved after discussion and it was agreed that S04.40XA was the correct code.

After the review process, the coders achieved high degrees of agreement for both Inpatient and Profee coding.

4 Dataset Analysis

In this section, we present various statistics of MDACE, including the number of annotated encounters/charts, documents, unique codes and evidence spans (Table 2). Since annotating all evidence is more time consuming than annotating sufficient evidence, the Profee coders only completed a small subset (52) of the Inpatient charts.

Tables 3 and 4 show the distribution of evidence spans in different note categories. Research on deep learning models for CAC has been mostly focused on using discharge summaries for code prediction. The tables show that although discharge summaries capture the majority of coding related narratives for Inpatient, they are insufficient for Profee coding. Other notes, such as Physician and Radiology notes, should also be used.

Note Category	Evidence Count	Percentage
Discharge Summary	3,434	87.3
Physician	364	9.3
Radiology	60	1.5
General	28	0.7
Nutrition	19	0.5
Respiratory	12	0.3

Table 3: Distribution of evidence spans in Inpatient notes (cutoff at 10)

Note Category	Evidence Count	Percentage
Physician	2,082	37.4
Discharge Summary	1,584	28.5
Radiology	1,269	22.8
ECG	256	4.6
Echo	207	3.7
Rehab Services	66	1.2
General	29	0.5
Respiratory	27	0.5
Nutrition	26	0.5

Table 4: Distribution of evidence spans in Profee notes (cutoff at 10)

Table 5 shows the overlap between the MIMIC codes and MDACE codes[4]. There is less than 50% code overlap, indicating that a high percentage of MIMIC codes are missing from our annotations. There are two possible explanations for this: firstly, over 37% of the 302 MIMIC encounters are missing operative notes, and as a result, the coders could not annotate the procedure codes which account for 33% of the missing Inpatient codes; and secondly, coding guidelines have changed over the years, and our coders were likely following different coding standards from the MIMIC coders; However, verifying such a claim without information about the MIMIC coding process is impossible. It should be noted that a similar observation of low agreement with MIMIC coders based on 508 re-annotated discharge summaries was also reported in (Kim and Ganapathi, 2021). Our coders added an average of 25 extra codes per chart for Profee coding, a result of their effort to annotate all evidence spans.

Table 6 summarizes the mapping from ICD-10 to ICD-9 codes. The majority of the mappings, 92% for Inpatient and 87% for Profee, were either verified by coders during the annotation process or based on a single identical or approximate match in GEMs. This gives us high confidence with the quality of the mapped ICD-9 codes.

[4] We ignored CPT codes for Evaluation and Management (E&M) which are in the range of 99201 and 99499 as they require a decision making calculator to arrive at the correct CPT codes rather than simply depending on the clinical text.

Codes	Inpatient	Profee
MIMIC	5,250	694
MDACE	3,414	1,630
Agreed	2,370 (45.1%)	306 (44.1%)
Missed	2,880 (54.9%)	388 (55.9%)
Added (average)	3.457	25.462

Table 5: Comparison of MIMIC-III and MDACE codes

ICD-10 to ICD-9	Inpatient	Profee
Coder Verified	2,525 (64.2%)	1,606 (28.9%)
Identical match	417 (10.6%)	1,387 (24.9%)
Approximate match	687 (17.5%)	1,847 (33.2%)
Multiple match	244 (6.2%)	704 (12.6%)
Other	61 (1.6%)	19 (0.3%)

Table 6: Distribution of ICD-10 to ICD-9 code mappings

5 Evidence Extraction Methods

This section introduces several evidence extraction methods that we implemented within a convolutional neural network based model to establish baselines for code evidence extraction on MDACE.

5.1 EffectiveCAN

EffecitiveCAN (Liu et al., 2021) is a convolution-based multi-label text classifier that achieved state-of-the-art performance on ICD-9 code prediction on MIMIC-III. It encodes the input text through multiple layers of residual squeeze-and-excitation (Res-SE) convolutional block to generate informative representations of the document. It uses label-wise attention to generate label specific representations, which has been widely used by DL models to improve predictions as well as to provide an explanation mechanism of the model, e.g., (Mullenbach et al., 2018). We chose EffectiveCAN as our base model for its simplicity, efficiency, and high performance. Its attention weights can be viewed as soft masks, making it a natural fit for producing baseline evidence results on MDACE.

5.2 Evidence Extraction Methods

We implemented multiple baseline approaches for code evidence extraction, including unsupervised attention, supervised attention, linear tagging and CNN tagging. Figure 1 shows our implementation of the EffectiveCAN model with additional attention supervision mechanism for evidence extraction.

5.2.1 Unsupervised Attention

EffectiveCAN uses text encoding from multiple layers of Res-SE block to generate key for the at-

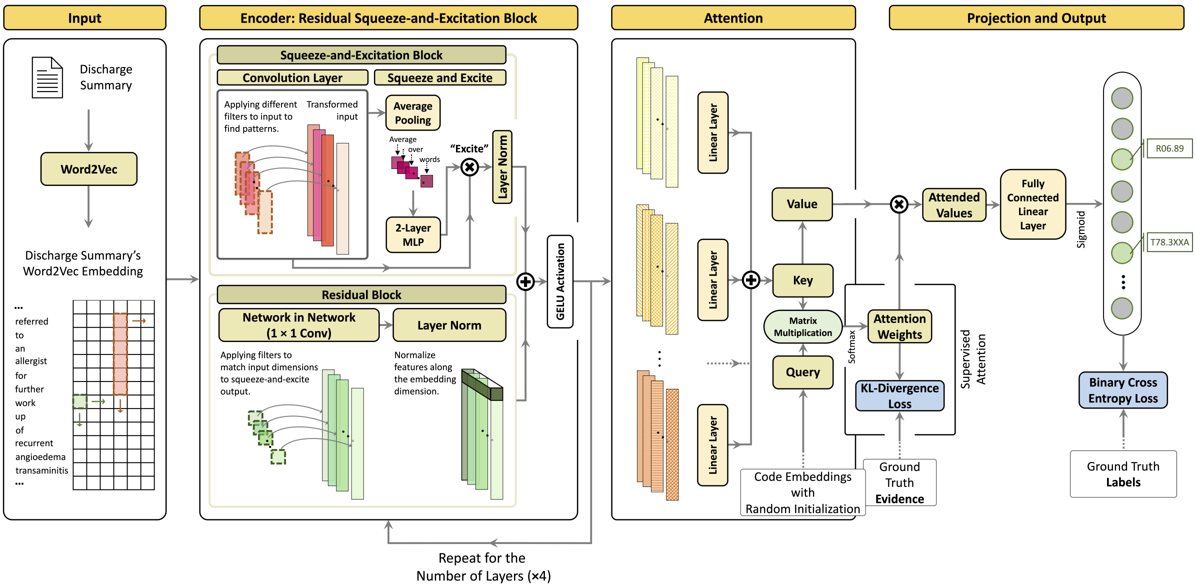

Figure 1: The architecture of EffectiveCAN with supervised attention.

tention module. The result is a label-specific representation of the input obtained by multiplying the key (value) by the attention weights. The attention weights signal the most relevant parts of the input text with respect to the output, and can be used to extract evidence for the predicted codes. We consider this the simplest baseline and compare the performance of other supervised methods with it.

5.2.2 Supervised Attention (SA)

We added a loss for evidence supervision during training as illustrated in Equation 1. We chose Kullback–Leibler (KL) divergence loss over other losses such as mean squared error because it is a term in the cross-entropy loss expression, and would result in a similar gradient behavior to the binary-cross entropy (BCE) loss used for code prediction (Yu et al., 2021).

$$\mathcal{L} = \mathcal{L}_{BCE}(\hat{\mathbf{y}}_{code}, \mathbf{y}_{code}) + \lambda_1 \, \mathcal{L}_{KLD}(\mathbf{a}, \mathbf{y}_{evd}) \tag{1}$$

where **a** is the attention weights.

5.2.3 Linear Tagging Layer

Inspired by the work of Segal et al. (2020) on the use of tagging for question answering, we added a feed-forward tagging layer, f_{tag}, on top of EffectiveCAN for evidence extraction. We use the output of the last Res-SE block, $\mathbf{h}^l$, and the normalized attention scores w.r.t. the maximum weight, $\mathbf{a}_{scaled}$, as inputs to two linear layers that share parameters for all the labels. The scaling is done so that the maximum score would be consistent among different instances. The outputs of these linear layers are multiplied to obtain the logits for evidence prediction, $\hat{\mathbf{y}}_{evd} \in \mathbb{R}^N$ (where N is the text length and each token is labeled as evidence or not). We used BCE for the tagging loss, and added it to the label loss through a weight term:

$$\hat{\mathbf{y}_{evd}} = \sigma\big(f_1(\mathbf{h}^{l=4}) \times f_2(\mathbf{a}_{scaled})\big) \tag{2}$$

$$\mathcal{L} = \mathcal{L}_{BCE}(\hat{\mathbf{y}}_{code}, \mathbf{y}_{code}) + \lambda_2 \, \mathcal{L}_{BCE}(\hat{\mathbf{y}}_{evd}, \mathbf{y}_{evd}) \tag{3}$$

5.2.4 CNN Tagging Layer

We extended the linear tagging layer by adding a CNN layer as another method for evidence extraction. The CNN tagger, f_{cnn}, has as input the sum of the two linear layer projection of the last Res-SE block, the normalized attention scores, and the code embeddings, **u**. The inputs are then fed into a 1-D convolutional layer ($conv1D$) with kernel size of 9 and out-channel size of 10, followed by layer normalization, ReLU activation, and finally a linear layer (f_4) to project the output back to the original dimension.

$$\mathbf{x} = f_1(\mathbf{h}^{l=4}) + f_2(\mathbf{a}_{scaled}) + f_3(\mathbf{u}) \tag{4}$$

$$\hat{\mathbf{y}}_{evd} = \sigma\big(f_4(conv1D(\mathbf{x}))\big) \tag{5}$$

The output logits from the final layer are used for evidence prediction, with the same BCE loss as the linear tagger, shown in Equation 3.

Train Set	Code-F1	Token-F1
0	0.583	0.320
$^{1}/_{8}$	0.581	0.323
$^{1}/_{4}$	0.577	0.328
$^{1}/_{2}$	0.582	0.332
$^{3}/_{4}$	0.581	0.362
1	0.581	0.368

Table 7: Supervised attention training performance on dev set for evidence training datasets of different sizes.

Data Splits	Train	Dev	Test
Code (c)	c.train	c.dev	c.test
	47,719	1,631	3,372
Evidence (ev)	ev.train	ev.dev	ev.test
Inpatient	181	60	61
Profee	31	10	11
Code+Evidence	c.train	c.dev	c.test - ev.dev
	+ ev.train	+ ev.dev	- ev.train
Inpatient	47,900	1,691	3,131
Profee	47,750	1,641	3,331

Table 8: Our new Code+Evidence data splits based on the splits of Mullenbach et al. (2018) for code prediction and our evidence dataset splits.

6 Experiments and Results

In this section, we describe the experiments for evaluating the evidence extraction methods introduced in Section 5, using the token- and span-level metrics in Section 6.2.

6.1 Data Splits

Rather than simply making random train/dev/test splits, we created sub-training splits to effectively determine the optimum splits for low resource semi-supervised evidence learning. We randomly sampled fixed development and test sets with 10% of the annotated charts (overall, 20% was held out). Next, we used different portions of the remaining 80% data to create 12.5%, 25%, 50%, 75%, and 100% training sets to train the attention weights of the EffectiveCAN model as shown in Table 7. As a result, we established the data size needed for supervised training, while the remaining data can be used to create a more representative test set. For results given in Table 7, $\lambda_1 = 0.5$ was used as the hyperparameter in Equation 1 without any hyperparameter tuning.

We decided to use the 75% split point since the evidence training showed only slight improvement with more data. Hence, the created evidence data splits are 60%/20%/20% for train/dev/test. The new data splits for code and evidence are given in Table 8[5]. We adopted the train/dev splits (c.train and c.dev) of Mullenbach et al. (2018) for code prediction as they have been widely used for comparing the performance of deep learning models. We removed the evidence train and dev examples (ev.train and ev.dev) from their test set (c.test) so as to follow the standard data use practices.

Table 7 also shows that adding labeled evidence data to the code train/dev sets did not affect code prediction significantly. This is reasonable given that the evidence dataset is much smaller than the code dataset. Compared with the results in (Liu et al., 2021), we can see that the code prediction F1 does not change significantly with or without evidence training. This means that code prediction performance established on the Mullenbach et al. (2018) data splits can be transferred to the MDACE data splits without much concern.

6.2 Evaluation Metrics

We evaluate the evidence extraction methods using the precision, recall, and micro-F1 score on five main metrics: Token match, Exact span match, Partial span match, Position independent (P.I.) token match, and P.I. exact span match. The token match metrics are used to measure the predicted evidence label of each token in a document compared to its ground truth label. The span metrics measure the whole evidence span, which is defined as consecutive tokens with the evidence label. An exact span match considers complete overlap with the ground truth span as correct, while partial span match treats any overlap as correct. These metrics measure how well the evidence extraction methods generate whole spans rather than disjoint, correct tokens. The P.I. metrics disregard the location of the evidence span/token and consider an evidence as correct based on string matching. These metrics are used to alleviate the issue of sufficient vs. complete evidence annotation explained in Section 3.1.

We use the model's precision-recall curve on the dev set to determine a threshold that maximizes the token match micro-F1 score, and use this threshold for evaluation on the test set.

6.3 Results

The evaluation results of the various evidence extraction methods on the discharge summaries of MDACE are shown in Table 9. The results for each method/model are from a single run of training. The EffectiveCAN based models have about 17 mil-

[5] Four records in the code training set were removed because they do not contain any billing codes.

Model	Threshold	Token Match			Exact Span Match			Partial Span Match			P.I. Token Match			P.I. Exact Span Match		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
CAML																
Unsup. Attention	0.02	3.81	10.7	5.62	0.43	2.13	0.71	4.18	20.9	6.97	6.73	18.0	9.80	0.88	4.16	1.45
EffectiveCAN																
Unsup. Attention	0.08	40.9	34.3	37.3	20.4	36.8	26.3	41.4	74.6	53.3	67.6	37.9	48.5	35.0	40.7	37.6
Sup. Attention	0.06	41.5	**45.3**	**43.3**	**22.1**	**45.0**	**29.6**	**40.2**	**82.0**	**54.0**	**68.1**	50.1	57.8	**36.7**	**49.6**	**42.2**
Linear Tagging	0.15	**41.9**	41.0	41.5	21.7	40.0	28.1	**40.2**	74.0	52.1	66.1	46.8	54.8	34.4	45.9	39.3
CNN Tagging	0.25	36.8	51.5	42.9	19.7	38.5	26.1	40.0	78.2	53.0	56.3	**60.2**	**58.2**	30.1	48.7	37.2

Table 9: Evaluation results of evidence extraction methods on the IP discharge summary test set of MDACE.

Dataset	Threshold	Token Match			Exact Span Match			Partial Span Match			P.I. Token Match			P.I. Exact Span Match		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Inpatient	0.06	34.9	40.2	37.4	17.3	38.8	23.9	33.2	74.6	46.0	68.0	46.2	55.1	35.4	45.6	39.9
Profee	0.02	41.0	50.9	45.4	28.2	49.2	35.9	42.4	74.0	54.0	51.7	61.0	56.0	29.7	55.8	38.8

Table 10: Evaluation results of the supervised attention model on the code-able notes test set of MDACE.

lion parameters, and each took about six hours to train on an AWS p3.2xlarge 1 GPU EC2 instance.

Out of all the evidence extraction methods tested, Supervised Attention performed the best across all metrics. The tagging methods under-performed SA, likely because they need more data to tune their parameters. Our experiments on a much larger proprietary evidence dataset showed that Linear Tagging outperformed SA. So the best evidence extraction methods could be based on the size of the training data.

We provide the performance of CAML's attention-based explanation (Mullenbach et al., 2018) for comparison. It should be noted that we used the filter size of 11 rather than 10 for training the CAML model, and the best micro-F1 we obtained is 0.525, lower than the F1 value of 0.539 as reported in the paper.

Since supervised attention resulted in better performance than other methods on discharge summaries, we used it to evaluate the effect of adding other code-able notes including physician and radiology notes to the input (Table 10). For training the model on Inpatient and Profee datasets, the maximum length for truncating text was increased from $3,500$ to $5,000$. Table 10 shows the performance of Inpatient vs. Profee coding. The position sensitive span metrics on Profee are significantly higher than those of Inpatient, likely the result of complete evidence annotations, as the gain was reduced on position-independent metrics.

We determined threshold values based on the token match metric for its simplicity. But we also take into consideration the other metrics, such as exact span match, to have a better grasp of how well the extracted evidence matches human annotations. Note that partial span match should be used with caution since if all the tokens were predicted as evidence, it would yield a perfect performance. Position independent token match takes tokens out of their context, which may result in evidence that is not reasonable to humans, e.g., "hr" where it means hour instead of heart rate.

The λ values in the loss Equations 1 and 3 were tuned such that the micro-F1 value for the code prediction task would remain close to the baseline value. For SA, 2.5 and 5.0 were considered for the λ coefficient, and $\lambda_1 = 2.5$ yielded code micro-F1 of 0.585, close to the baseline value of 0.584. For the tagging models, three values, 0.5, 1.0 and 2.0, were considered, and $\lambda_2 = 0.5$ yielded code micro-F1 of 0.583, close to the baseline performance for CNN tagging. These values were used for the reported results. For evidence prediction threshold, steps of 0.02 and 0.05 were used to generate the precision-recall curve for the attention-based and tagging methods respectively, and the threshold values are reported in Tables 9 and 10.

Table 11 in Appendix A provides example outputs from two baseline extraction methods.

7 Conclusions

In this paper, we introduce MDACE, the first publicly available code evidence dataset built on a subset of the MIMIC-III clinical records. The dataset contains evidence spans for diagnosis and procedure codes annotated by professional medical coders. MDACE addresses a critical need for CAC research to be able to automatically evaluate the code evidence generated by ML models. We believe that its release will greatly improve the understanding and application of deep learning technologies for medical coding and document classification.

8 Limitations

Professional coders are trained to find sufficient, as opposed to exhaustive, evidence for each code. Our Profee coders were instructed to find all the evidence for each code. However, given the large number of notes in some MIMIC encounters, they might only manage to annotate most of the evidence. For Inpatient, there might be more bias among coders towards finding sufficient evidence: namely, there were many cases in which one coder found evidence that another had not, but during the adjudication process, both coders agreed it should be included. Thus, although we have opened the door to automatic evaluation of evidence extraction systems, some metrics, such as recall on our dataset, might underestimate the true recall of a system.

We discovered many human errors while cleaning up the data, including wrong codes and partial highlights. We tried our best to fix these issues, but some errors likely remain in the dataset.

References

Center for Medicare & Medicaid Services. 2009. General equivalence mappings: ICD-9-CM to and from ICD-10-CM and ICD-10-PCS. `https://library.ahima.org/PdfView?oid=92359`.

Nils Feldhus, Robert Schwarzenberg, and Sebastian Möller. 2021. Thermostat: A large collection of nlp model explanations and analysis tools. *arXiv preprint arXiv:2108.13961*.

Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Shuyuan Hu, Fei Teng, Lufei Huang, Jun Yan, and Haibo Zhang. 2021. An explainable CNN approach for medical codes prediction from clinical text. *BMC Medical Informatics and Decision Making*.

Chao-Wei Huang, Shang-Chi Tsai, and Yun-Nung Chen. 2022. PLM-ICD: Automatic ICD coding with pretrained language models. *arXiv preprint arXiv:2207.05289*.

Shaoxiong Ji, Wei Sun, Hang Dong, Honghan Wu, and Pekka Marttinen. 2022. A unified review of deep learning for automated medical coding. *arXiv preprint arXiv:2201.02797*.

Alistair Johnson, Lucas Bulgarelli, Tom Pollard, Steven Horng, Leo Anthony Celi, and Roger Mark. 2020. "MIMIC-IV" (version 0.4). *PhysioNet*. `https://mimic.mit.edu/docs/iv/about/`.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Byung-Hak Kim and Varun Ganapathi. 2021. Read, attend, and code: Pushing the limits of medical codes prediction from clinical notes by machines. In *Proceedings of the 6th Machine Learning for Healthcare Conference*, volume 149 of *Proceedings of Machine Learning Research*, pages 196–208. PMLR.

Klaus Krippendorff. 2004. *Content analysis: An introduction to its methodology (2nd Edition*. CA: Sage Publications.

Leibo Liu, Oscar Perez-Concha, Anthony Nguyen, Vicki Bennett, and Louisa Jorm. 2022. Hierarchical label-wise attention transformer model for explainable ICD coding. *arXiv preprint arXiv:2204.10716*.

Yang Liu, Hua Cheng, Russell Klopfer, Matthew R. Gormley, and Thomas Schaaf. 2021. Effective convolutional attention network for multi-label clinical document classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5941–5953, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*.

Damian Pascual, Sandro Luck, and Roger Wattenhofer. 2021. Towards BERT-based automatic ICD coding: Limitations and opportunities. *Proceedings of the BioNLP 2021 Workshop*, pages 54–63.

Danish Pruthi, Bhuwan Dhingra, Graham Neubig, and Zachary C. Lipton. 2020. Weakly- and semi-supervised evidence extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3965–3970, Online. Association for Computational Linguistics.

Thomas Searle, Zina Ibrahim, and Richard Dobson. 2020. Experimental evaluation and development of a silver-standard for the MIMIC-III clinical coding dataset. In *Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing*, pages 76–85, Online. Association for Computational Linguistics.

Elad Segal, Avia Efrat, Mor Shoham, Amir Globerson, and Jonathan Berant. 2020. A simple and effective model for answering multi-span questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3074–3080, Online. Association for Computational Linguistics.

Cansu Sen, Bingyang Ye, Javed Aslam, and Amir Tahmasebi. 2021. From extreme multi-label to multi-class: A hierarchical approach for automated ICD-10 coding using phrase-level attention. *arXiv preprint arXiv:2102.09136*.

Wu-Chen Su, Kevin Dufendach, and Danny Wu. 2019. Assessing the readability of freely available ICU notes. *Proceedings of the AMIA Joint Summits on Translational Science*.

Thanh Vu, Dat Quoc Nguyen, and Anthony Nguyen. 2020. A label attention model for ICD coding from clinical text. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)*.

Sarah Wiegreffe and Ana Marasovic. 2021. Teach me to explain: A review of datasets for explainable NLP. *CoRR*, abs/2102.12060.

Xiancheng Xie, Yun Xiong, Philip Yu, and Yamgyong Zhu. 2019. EHR coding with multi-scale feature attention and structured knowledge graph propagation. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*.

Mo Yu, Yang Zhang, Shiyu Chang, and Tommi Jaakkola. 2021. Understanding interlocking dynamics of cooperative rationalization. In *Advances in Neural Information Processing Systems*, volume 34, pages 12822–12835. Curran Associates, Inc.

Zheng Yuan, Chuanqi Tan, and Songfang Huang. 2022. Code synonyms do matter: Multiple synonyms matching network for automatic ICD coding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 808–814, Dublin, Ireland. Association for Computational Linguistics.

Ruiqi Zhong, Steven Shao, and Kathleen McKeown. 2019. Fine-grained sentiment analysis with faithful attention. *arXiv preprint arXiv:1908.06870*.

A Examples of Generated Evidence

Examples of predicted evidence, using unsupervised attention weights as the baseline and the supervised attention method, are given in Table 11.

Code	Human Annotation	Baseline	Supervised Attention	Code description
36.15	“left internal mammary artery to left anterior descending artery”	“mammary”	“left internal mammary” “left anterior descending”	Single internal mammary-coronary artery bypass
427.31	“Atrial fibrillation”	“Atrial” ×2 “atrial”	“Atrial fibrillation” ×2	Atrial fibrillation
424.1	“aortic stenosis”	“Sj”	“Sj” “Aortic (aortic ×2)″	Aortic valve disorders
441.2	“thoracic aortic aneurysm”	“thoracic”	“thoracic” “aneurysm”	Thoracic aneurysm without mention of rupture
428.0	“Congestive heart failure”	“Congestive” ×2	“Congestive heart failure” ×2	Congestive heart failure, unspecified
790.92	“Supratherapeutic INR”	“INR” ×3	“INR” “Supratherapeutic INR”	Abnormal coagulation profile
584.9	“Acute Renal Failure”	“Renal” “creatinine” “renal”	“Acute Renal Failure” “renal failure”	Acute kidney failure, unspecified
585.9	“Chronic renal insufficiency”	“renal” ×2	“renal insufficiency” ×2	Chronic kidney disease, unspecified

Table 11: Examples of generated evidence

B Annotation Guidelines

The task is to annotate MIMIC charts with sufficient code evidence based on the documentations using an open source tool called INCEpTION.

- For Inpatient coding, annotate evidence for ICD-10-CM and ICD-10-PCS codes.
- For Profee coding, annotate evidence for ICD-10-CM and CPT codes (ignoring EM codes which are in the range of 99201-99499).

Reference the latest coding book to decide whether an ICD-10 code is supported by the documentation. If there is a definitive diagnosis, do not code symptom codes, otherwise symptom codes can be coded. Code external cause codes only with injury codes.

Code as in real life, once a condition is confirmed and you feel comfortable with a code assignment, annotate the text spans with the code and move on to the next one. You are encouraged to provide multiple evidence for a code, as long as it doesn't slow you down too much. For Profee coding, go through all notes and annotate as many diagnoses as possible.

The general annotation process includes:

- Leaf through chart documents to find the ones appropriate to code from. Highlight best/sufficient text spans as evidence for a code.
- Choose the appropriate ICD-10/ICD-9 code pair or CPT code in the Label box to assign to the highlighted text span.
- If the correct ICD-10 or CPT code is not in the label set, type it in the Label box and assign it to the highlighted text span.
- Try to annotate evidence for all ICD-9 or CPT codes in the label set if there is supporting documentation.

Follow these instructions to annotate and export a chart in INCEpTION:

1. Go to Dashboard and click Annotation, select a document to open.
2. Select Search in the left panel. You can search any phrase and select the document containing the phrase to annotate.
3. Open the Preferences popup, and set the following (Done once for a project):
   - Editor: brat (line-oriented)
   - Sidebar right: 30
   - Page size: 1000
4. In a document, double click on a word or highlight a text span, and then select a label from the right panel. You can also start typing in the label box and the matching labels will show up.
5. You can navigate through the documents using the icons at the top of the middle panel, and move through the annotations using the arrows in the right panel.
6. After you finish annotating a chart, select Administration -> MIMIC-encounterID -> Settings -> Export, choose WebAnno TSV v3.3 format and then Export the whole project.

These code evidence annotations will be made available to the research communities so those with access to the MIMIC dataset can use them to evaluate the code evidence generated by their ML models.