Benchmarking Commercial Intent Detection Services with Practice-Driven Evaluations

Haode Qi†[*], Lin Pan†[*], Atin Sood†, Abhishek Shah†
Ladislav Kunc†, Mo Yu‡, Saloni Potdar†
†IBM Watson
‡MIT-IBM Watson AI Lab
{Haode.Qi,Abhishek.Shah1,lada}@ibm.com
{panl,asood,yum,potdars}@us.ibm.com

Abstract

Intent detection is a key component of modern goal-oriented dialog systems that accomplish a user task by predicting the intent of users' text input. There are three primary challenges in designing robust and accurate intent detection models. First, typical intent detection models require a large amount of labeled data to achieve high accuracy. Unfortunately, in practical scenarios it is more common to find small, unbalanced, and noisy datasets. Secondly, even with large training data, the intent detection models can see a different distribution of test data when being deployed in the real world, leading to poor accuracy. Finally, a practical intent detection model must be computationally efficient in both training and single query inference so that it can be used continuously and retrained frequently. We benchmark intent detection methods on a variety of datasets. Our results show that Watson Assistant's intent detection model outperforms other commercial solutions and is comparable to large pretrained language models while requiring only a fraction of computational resources and training data. Watson Assistant demonstrates a higher degree of robustness when the training and test distributions differ.

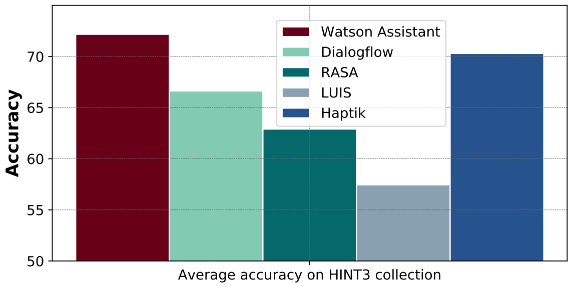

Figure 1: **Accuracy of commercial solutions** on the HINT3 collection of datasets. Results are averaged across the *Full* versions of the three datasets and their *Subset* versions. The in-scope accuracy is reported on a threshold of 0.1. Watson Assistant achieves the best results on average. Results for all methods except Watson Assistant are obtained from Arora et al. (2020).

1 Introduction

Intent detection and entity recognition form the basis of the Natural Language Understanding (NLU) components of a task-oriented dialog system. The intents and entities identified in a given user utterance help trigger the appropriate conditions defined in a dialog tree which guides the user through a predetermined dialog-flow. These task-oriented dialog systems have gained popularity for designing applications around customer support, personal assistants, and opinion mining, etc.

The Conversational AI market is expected to grow to an estimated USD 13.9 billion by 2025 as reported by Markets & Markets [1]. There are several solutions in the market that help enterprises build and deploy chatbots quickly to automate large portions of their customer service interactions. Hence, a commercial conversational AI solution needs to adapt to a variety of use cases, accurately identify users' intents and resolve their queries.

There are three primary challenges in designing intent detection models that power real-world dialog systems: (1) **Limitations in training data**: while typical machine learning models are trained on large, balanced, labeled datasets, practical intent detection systems rely on customer provided data. These datasets are usually small, probably noisy, unbalanced, and contain classes with overlapping semantics, etc. The relatively poor quality of training data makes it hard to train accurate models. (2) **Robustness to non-standard user inputs**: when the intent detection models are deployed in real-world settings, they often operate on test data that differs significantly from the training data. The mismatch in train and test data distributions mainly

[*] Equal contribution

[1] https://customerthink.com/conversational-ai-in-2021-3-top-trends-to-look-out-for

Proceedings of NAACL HLT 2021: IndustryTrack Papers, pages 304–310
June 6–11, 2021. ©2021 Association for Computational Linguistics

comes from the free-form of input user queries. These real world queries express the same intents with their non-standard paraphrases, which are difficult to fully cover during training. The lack of large and clean training data makes this problem worse. (3) **Computational efficiency**: the intent detection models should be computationally efficient for both training and inference. On one hand, efficient inference is crucial since it allows for faster query resolution times for the users.[2] On the other hand, a real-world dialog system is frequently updated according to customer needs, so faster training time becomes an important consideration for real-world conversational AI solutions.

In this work, taking the aforementioned three realistic challenges into consideration, we evaluate multiple intent detection models and focus on their accuracy, data efficiency, robustness, and computational efficiency. We compare the performance of various commercial intent detection models on three datasets in the HINT3 collection (Arora et al., 2020). We also evaluate pretrained Language Models (LM) on three commonly used public datasets for benchmarking intent detection - CLINC150 (Larson et al., 2019), BANKING77 (Casanueva et al., 2020), and HUW64 (Liu et al., 2019b). In addition, we create few-shot learning settings from these datasets, to better match real world low-resource scenarios. Furthermore, we measure the "in the wild" robustness of the systems via creating difficult test subsets from existing test sets. Finally, we evaluate the classification accuracy and training time of these models because it directly affects the usability and development lifecycle of an conversational AI solution.

We build upon the existing study in Arora et al. (2020) which benchmarked commercial solutions aside from IBM Watson Assistant (i.e., Dialogflow, LUIS, and RASA). We extend this study by adding Watson Assistant and recent large-scale pretrained LMs. We also explore few-shot and robustness settings, and compare the resource efficiency and training times of different commercial solutions as well as pretrained LMs. Among these solutions, Watson Assistant's new intent detection algorithm performs better than other commercial solutions (Figure 1), and achieves comparable accuracy when compared to large-scale pretrained LMs (Figure 2) while being much more efficient.

2 Related Work

Several datasets have been released to test the performance of intent detection for task-oriented dialog systems such as Web Apps, Ask Ubuntu and Chatbot corpus from Braun et al. (2017); ATIS dataset (Price, 1990) and SNIPS (Coucke et al., 2018). The ATIS and SNIPS datasets have been created with focus on voice interactive chatbots. Voice modality has some specific characters, i.e., it does not contain typos and it is less noisy than text-based communication. Thus, these datasets are oversimplified version of the text-based intent detection task "in the wild" due to well-constructed dataset and limited number of classes.

Recently, CLINC150 (Larson et al., 2019), BANKING77 (Casanueva et al., 2020), and HWU64 (Liu et al., 2019b) have been used to benchmark the performance of intent detection systems. These datasets cover a large number of intents across a wider range of domains, which captures more real-world complexity of doing fine-grained classification. Arora et al. (2020) proposed a new collection of datasets called HINT3, containing a noisy and diverse set of intents and examples across three domains sourced from domain experts and real users.

Prior work from Arora et al. (2020), Braun et al. (2017), and Liu et al. (2019a) study the performance of different conversational AI services using the datasets mentioned above. Casanueva et al. (2020), Larson et al. (2019), Arora et al. (2020), Bunk et al. (2020) and others have benchmarked several state-of-the-art (SOTA) pretrained LMs such as BERT (Devlin et al., 2019) on the aforementioned datasets.

We aim to standardize the benchmarking tests that need to be run while developing an industry scale intent detection system. The tests should cover a variety of real-world datasets, settings such as few-shot scenarios and testing on semantically dissimilar test examples. Additionally, the tests should also cover resource efficiency and training time - since they affect the overall deployment costs of a virtual assistant cloud service. A carefully chosen trade-off between accuracy and efficiency is the decision making factor in choosing the algorithm for the real-world intent detection system.

[2] Inference time is usually dependent on service-level agreements between the provider and the user which determine the response time upper bounds of the APIs. This is hard to measure and compare across services in a reliable way for the purpose of this study.

3 Evaluation Settings

3.1 Datasets

We create our proposed evaluation settings based on the following public intent detection datasets:

CLINC150 consists of $22,500$ in-scope examples that cover 150 intents in 10 domains, such as banking, work, travel, etc. The dataset also comes with $1,200$ out-of-scope examples. In this work, we only focus on the in-scope examples.

HWU64 contains $25,716$ examples, covering 64 intents in 21 domains. The data creation process aims to reflect human-home robot interaction. We are using one fold train-test split with $9,960$ training examples and $1,076$ testing examples.

BANKING77 is a single domain dataset created for fine-grained intent detection. It focuses on the banking domain, and has $13,083$ examples covering 77 intents.

3.2 Practice-Driven Benchmark Settings

Full-set setting This corresponds to the standard evaluation setting that uses the full training and testing sets.

Few-shot setting In the real-world setting, users may not provide a large number of labelled examples to train a conversational AI system. Labeling data is extremely time consuming and difficult, so we need to make our intent detection systems robust enough to handle the few-shot scenarios and improve time to value for the user. We create a few shot setup for all the datasets by sampling 5 examples per intent and 30 examples per intent on CLINC150, HWU64 and BANKING77 datasets.

Difficult test setting Most of the current SOTA classification models can achieve 90%+ test accuracy on the aforementioned public datasets. However this is due to the presence of a large number of similar and standard queries in the training and test set. To reflect the performance in realistic settings, where users can input non-standard paraphrases of the queries, we propose to create more difficult subsets of the provided test sets to mimic the real-world setting.

Following Arora et al. (2020), we create a subset of each test set with semantically dissimilar sentences from the training set. Instead of using ELMo (Peters et al., 2018) and entailment scores, we use TF/IDF cosine distance to pick the most difficult examples from the original test sets. Each intent is treated separately during the selection process. First, all training utterances in a specific intent are tokenized (using simple white-space based tokenizer, ignoring punctuation). These tokenized training utterances are concatenated and transformed to TF/IDF vector space. Then, each testing example of the intent is transformed using the initialized TF/IDF transformer and cosine similarities with the transformed training set are calculated. Finally, 5 least similar examples per intent are selected for inclusion to the difficult test set. For example, the CLINC150 dataset has 150 intents, so our algorithm creates a test set of 750 examples. Analogous process is used for the other two datasets. [3]

4 Experiment I: Comparison with Pretrained LMs

Pretrained LMs finetuned for intent detection have been shown to perform very well in recent literature, such as (Casanueva et al., 2020). Users can modify and adapt pretrained LMs to serve them as part of a scalable solution. However, this often requires a complex solution design, an example of which can be found in Yu et al. (2020). In this work we evaluate and compare the commercial services with the following pretrained LMs: **USE$_{\textbf{base}}$**, i.e., Universal Sentence Encoder (Cer et al., 2018); **Distilbert$_{\textbf{base}}$** (Sanh et al., 2020); **BERT$_{\textbf{base}}$**, **BERT$_{\textbf{large}}$** (Devlin et al., 2019); and **RoBERTa$_{\textbf{base}}$** (Liu et al., 2019b).

We compare Watson Assistant, RASA, and the aforementioned pretrained-LMs on the datasets and settings described in Section 3, and measure the training time as well as accuracy.

Watson Assistant We evaluate both the *classic* version of IBM Watson Assistant (WA) [4] and the *enhanced* version with improved intent detection algorithm. Public API is used to train and evaluate the model. For training time, we measure the round-trip latency from sending the training request until we receive the status that the model is trained and available for serving. [5]

[3] We release the difficult subsets at `https://github.com/haodeqi/BenchmarkingIntentDetection` to facilitate repeatability and future research.

[4] `https://www.ibm.com/cloud/watson-assistant`

[5] Note that the training times may vary depending on the load on the web API.

	CLINC150	HWU64	BANKING77	Average
WA classic	93.9	88.8	90.6	91.1
WA enhanced	95.7	90.5	92.6	92.9
RASA	89.4	84.9	89.9	88.1
Distilbert-base	96.3	91.7	92.1	93.4
BERT-base	96.8	91.6	93.3	93.9
BERT-large	**97.1**	91.9	93.7	94.2
USE-base	94.7	88.9	89.9	91.2
RoBERTa-base	97.0	**92.1**	**94.1**	**94.4**

Table 1: **Accuracy on CLINC150, HWU64 and BANKING77 for Watson Assistant (WA), RASA and pretrained LMs**. Training is performed on the full train sets and evaluation on full test sets.

RASA[6] The tool offers the flexibility to incorporate other open-source models such as Transformer-based (Vaswani et al., 2017) models into the pipeline. For our experiments, we use the default training setting that trains a count-based feature ensemble with the DIETClassifier (Bunk et al., 2020).

Pretrained LMs For BERT-based models, we add a softmax classifier on top of the [CLS] token and finetune all layers. We use AdamW (Loshchilov and Hutter, 2018) with 0.01 weight decay and a linear learning rate scheduler. We choose a batch size of 32, max sequence length 128 and learning rate warmup for the first 10% of the total iterations, peaking at 0.00004. For training set variants of 5/30/all examples per intent, we train for 50/18/5 epochs, respectively. For USE$_{\text{base}}$ model, we train a softmax layer on top of the sentence representation and finetune all layers for 100 epochs. A learning rate of 0.05 and batch size of 32 are used for all training set variants. All models are trained with a single CPU core and a single K80 GPU.

4.1 Results and Analysis

Results in the full-set setting Table 1 shows results of Watson Assistant, RASA and pretrained LMs on CLINC150, HWU64, and BANKING77. We train on the full training sets and report result on the full test sets, measured by accuracy. The overall best finetuned LM RoBERTa$_{\text{base}}$ achieves 1.5% higher accuracy than Watson Assistant *enhanced*. However, the improvement from finetuning large pretrained LMs requires more computational resources.

Results in the few-shot setting Table 2 shows results on few-shot setting for 5/30/all examples per intent on CLINC150, HWU64 and BANKING77 datasets on the full test sets. For experimental settings and dataset details, refer to Section 4.

	CLINC150					
	5 ex/class		**30 ex/class**		**full**	
	Training time	Accuracy	Training time	Accuracy	Training time	Accuracy
WA classic	0.58	78.1	0.78	90.3	1.04	93.9
WA enhanced	0.66	83.6	0.63	92.5	1.81	95.7
RASA	1.25	53.2	5.6	79.4	13.93	89.4
Distilbert-base	15.23	82.2	31.65	93.2	35.98	96.3
BERT-base	29.67	83.8	61.43	94.7	71.08	96.8
BERT-large	125	87	280	95.8	270	97.1
USE-base	1.63	83.9	6.5	92.9	14.73	94.7
RoBERTa-base	33	86.3	85	95.4	90	97.0
	HWU64					
	5 ex/class		**30 ex/class**		**full**	
	Training time	Accuracy	Training time	Accuracy	Training time	Accuracy
WA classic	0.39	64.1	0.59	81.4	0.85	88.8
WA enhanced	0.75	71.0	0.54	86.2	0.82	90.5
RASA	0.67	43.7	2.17	72.4	9.43	84.9
Distilbert-base	6.32	71.1	13.92	86.3	20.35	91.7
BERT-base	12.73	70.1	27.18	87.5	39.48	91.6
BERT-large	52	77.3	120	89.3	175	91.9
USE-base	1.2	72.5	2.46	86.3	8.92	88.9
RoBERTa-base	13	71.7	40	88.8	60	92.1
	BANKING77					
	5 ex/class		**30 ex/class**		full	
	Training time	Accuracy	Training time	Accuracy	Training time	Accuracy
WA classic	0.38	64.2	0.49	84.7	0.64	90.6
WA enhanced	0.65	69.9	0.52	87.0	1.22	92.6
RASA	0.89	45.1	3.67	81.6	15.45	89.9
Distilbert-base	7.87	69.8	16.83	87.8	20.35	92.1
BERT-base	15.23	68.3	32.72	88.9	38.75	93.3
BERT-large	92	71.2	210	89.9	175	93.7
USE-base	1.33	65.3	2.95	86.8	9.47	89.9
RoBERTa-base	17	75.9	42	90.4	57	94.1

Table 2: **Accuracy and training time (in minutes) comparing Watson Assistant (WA) with RASA and pretrained LMs**. We use 5/30/all examples per intent on CLINC150, HWU64 and BANKING77 datasets. Results are the on the respective full test sets.

	CLINC150	HWU64	BANKING77
WA classic	79.3	83.4	75.2
WA enhanced	86.0	85.8	80.6
RASA	68.3	78.9	76.9
Distilbert-base	85.7	87.4	79.2
BERT-base	87.6	87.6	81.7
BERT-large	**89.5**	**89.2**	**83.9**
USE-base	81.6	83.4	74.5
RoBERTa-base	88.4	88.5	83.8

Table 3: **Accuracy on CLINC150, HWU64 and BANKING77 for Watson Assistant (WA), RASA and pretrained LMs**. Models are trained on full train sets and evaluated on Tfidf-difficult test sets.

Results in the difficult test setting Table 3 shows results on our difficult test sets. We observe that there is a significant drop in accuracy compared to the full test set, going from 90%+ to 80%s. This shows that these test sets are indeed more difficult for all algorithms, and they provide a better testbed for identifying the robustness of a

[6] `https://rasa.com`

	CLINC150					
	5 ex/class		**30 ex/class**		full	
	Training time	Accuracy	Training time	Accuracy	Training time	Accuracy
WA classic	0.58	54.0	0.78	69.9	1.04	79.3
WA enhanced	0.66	65.1	0.63	76.7	1.81	86.0
RASA	1.25	29.6	5.6	52.5	13.93	68.3
Distilbert-base	15.23	63.4	31.65	76.8	35.98	85.7
BERT-base	29.67	64.6	61.43	81.1	71.08	87.6
BERT-large	125	72.0	280	85.6	270	89.5
USE-base	1.63	66.6	6.5	77.5	14.73	81.6
RoBERTa-base	33	70.8	85	83.7	90	88.4
	HWU64					
	5 ex/class		**30 ex/class**		full	
	Training time	Accuracy	Training time	Accuracy	Training time	Accuracy
WA classic	0.39	53.9	0.59	72.3	0.85	83.4
WA enhanced	0.75	62.7	0.54	80.0	0.82	85.8
RASA	0.67	34.5	2.17	63.5	9.43	78.9
Distilbert-base	6.32	63.4	13.92	79.7	20.35	87.4
BERT-base	12.73	61.6	27.18	82.1	39.48	87.6
BERT-large	52	71.1	120	85.3	175	89.2
USE-base	1.2	66.3	2.46	79.8	8.92	83.4
RoBERTa-base	13	64.5	40	83.9	60	88.5
	BANKING77					
	5 ex/class		**30 ex/class**		full	
	Training time	Accuracy	Training time	Accuracy	Training time	Accuracy
WA classic	0.38	43.2	0.49	64.5	0.64	75.2
WA enhanced	0.65	49.1	0.52	69.7	1.22	80.6
RASA	0.89	26.9	3.67	57.9	15.45	76.9
Distilbert-base	7.87	50.0	16.83	69.0	20.35	79.2
BERT-base	15.23	48.3	32.72	73.4	38.75	81.7
BERT-large	92	52.6	210	75.8	175	83.9
USE-base	1.33	44.5	2.95	68.6	9.47	74.5
RoBERTa-base	17	57.1	42	75.7	57	83.8

Table 4: **Accuracy and training time (in minutes) comparing Watson Assistant (WA) with RASA and pretrained LMs**. We use 5/30/all examples per intent on CLINC150, HWU64 and BANKING77 datasets. Results are the on the respective Tfidf-difficult test sets.

intent detection system. In addition, we conduct the comparison in few-shot settings, where we use 5 examples per intent for training, and increase to 30 and full training sets. The complete set of results of few-shot setting on the difficult test sets can be found in Table 4. Results show that BERT$_{\text{large}}$ performs the best in terms of accuracy. However, Watson Assistant still stands on top considering the trade-off between training time and accuracy.

Training time vs accuracy trade-off We report the training times and resources used for all models across the three datasets in Table 5. We observe that the pretrained LMs require significantly more training time compared to Watson Assistant. For example, RoBERTa$_{\text{base}}$ achieves comparable performance to Watson Assistant but requires 90 minutes training time on CLINC150. Figure 2 shows a visualization of accuracy and training time for each model. Watson Assistant offers the best trade-off in terms of accuracy vs. training time.

We report results on HINT3 datasets for completeness and are discussed in Section 5 Table 8.

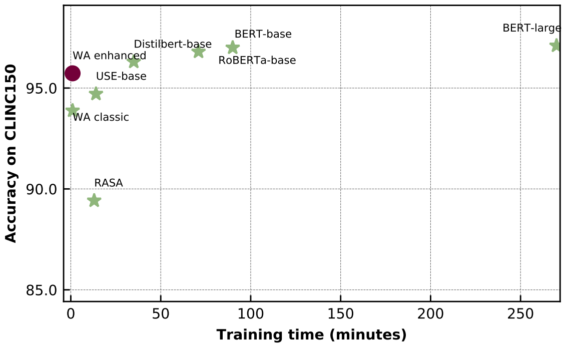

Figure 2: **Training time vs. accuracy on CLINC150** for Watson Assistant (WA), RASA and pretrained LMs. Full training set and test set are used. All methods except Watson Assistant are trained using GPU. Watson Assistant offers the best trade-off between training time and accuracy.

Algorithm	**Resources**	**CLINC150** Training time	**HWU** Training time	**BANKING77** Training time
WA classic	-	**1.04**	0.85	**0.64**
WA enhanced	-	1.81	**0.82**	1.22
RASA	GPU	13.93	9.43	15.45
Distilbert-base	GPU	35.98	20.35	20.35
BERT-base	GPU	71.08	39.48	38.75
BERT-large	GPU	270	175	175
USE-base	GPU	14.73	8.92	9.47
RoBERTa-base	GPU	90	60	57

Table 5: **Training time (in minutes) and resource requirements** for Watson Assistant (WA), RASA and pretrained LMs. Training is performed on full training sets. All methods except for Watson Assistant are trained using a single NVIDIA K80 GPU.

5 Experimental II: Comparison among Commercial Solutions

Finally, we conduct comparison studies among commercial services. Commercial solutions are more suitable for enterprise customers and are designed for users who have limited knowledge of machine learning and natural language processing. One of the challenges in comparing the performance of commercial services and designing experiments lies in the fact that most service providers have terms of use prohibiting any type of benchmarking on their services. To overcome this challenge, we use the prior benchmarking study from Arora et al. (2020) to obtain the performance of existing commercial solutions. In this benchmark, HINT3 dataset collection is used which contains three tasks with small amounts of training data. We extended the study by including the results on the Watson Assistant service.

In this section, we evaluate the perfor-

	SOFMattress		Curekart		Powerplay11	
	Train	Test (in-scope / out-of-scope)	Train	Test (in-scope / out-of-scope)	Train	Test (in-scope / out-of-scope)
Full	328	231/166	600	452/539	471	275/708
Subset	180	231/166	413	452/539	261	275/708

Table 6: **HINT3 training and test set statistics**. HINT3 consists of three datasets - SOFMattress, Curekart and Powerplay11. Each training set contains two versions - *Full* and *Subset*. The test set is also broken down into in-scope queries and out-of-scope queries.

mance of the following commercial solutions: **IBM Watson Assistant**[7], **Google Dialogflow**[8], **Microsoft LUIS**[9], and the open-source solution **RASA**[10]. We use the prior benchmarking study from Arora et al. (2020) to obtain the performance of these commercial solutions, except for Watson Assistant.

5.1 Datasets

HINT3 is a collection of three datasets: SOFMattress, Curekart, and Powerplay11. The statistics of the datasets are shown in Table 6. Each dataset has two training set variants referred to as *full* and *subset*. The subset variant was created by discarding semantically similar sentences using ELMo (Peters et al., 2018) and entailment score > 0.6 (Arora et al., 2020). We used both variants of the training data in our experiments. The test sets contain both in-scope and out-of-scope examples.

5.2 Experimental Setup

We use the same experimental setup as described in Arora et al. (2020). Following their methodology, we use a confidence threshold of 0.1. For the BERT model reported in their paper, they used $\text{BERT}_{\text{base}}$ and finetuned all layers upto 50 epochs, learning rate of 4×10^{-5} with warmup period of 0.1 and early stopping.

5.3 Results

Table 7 shows full results on the in-scope test examples of each dataset measured by accuracy using a confidence threshold of 0.1.

On average across the datasets (Table 8), Watson Assistant *enhanced* achieves 73.8% accuracy when trained on the full training sets and evaluated on the in-scope examples, outperforming DialogFlow by 4.57%, and LUIS by 13.87%. Training on the subset variant of the datasets, Watson Assistant also consistently outperforms the other commercial solutions. It is worth noting that Watson Assistant also does better than BERT by 4.4% on average.

	SOFMattress		Curekart		Powerplay11	
	full	subset	full	subset	full	subset
WA classic	73.6	66.2	83.2	79.9	63.3	57.1
WA enhanced	**74.0**	**68.4**	**86.7**	**85.4**	60.7	57.8
Dialogflow	73.1	65.3	75.0	71.2	59.6	55.6
RASA	69.2	56.2	84.0	80.5	49.0	38.5
LUIS	59.3	49.3	72.5	71.6	48.0	44.0
Haptik	72.2	64.0	80.3	79.8	**66.5**	**59.2**
BERT	73.5	57.1	83.6	82.3	58.5	53.0

Table 7: **In-scope Accuracy on HINT3 using commercial solutions.** We report the in-scope accuracy with a threshold of 0.1 for various intent detection methods. Results for all methods except Watson Assistant (WA) are obtained from (Arora et al., 2020).

	Full	Subset	Average
WA classic	73.4	67.7	70.6
WA enhanced	**73.8**	**70.5**	**72.2**
Dialogflow	69.2	64.0	66.6
RASA	67.4	58.4	62.9
LUIS	59.9	54.6	57.5
Haptik	73.0	67.6	70.3
BERT	71.9	64.1	68.0

Table 8: **Average In-scope Accuracy on HINT3 using commercial solutions.** We report the average in-scope accuracy across the three datasets with a threshold of 0.1 on Full and Subset versions of the HINT3 collection. Results for all methods except Watson Assistant (WA) are obtained from (Arora et al., 2020).

6 Conclusion

We proposed a new methodology to assess the performance of intent detection "in the wild" in task-oriented dialog systems. In practice, the platforms developed for building and deploying virtual assistants have to consider several scenarios and trade-offs. These systems have to train the best performing models in few-shot settings, strike a compromise between training time and accuracy, and adapt seamlessly to a wide range of domains.

We compare the performance of leading commercial services which are designed to develop task-oriented dialog systems on the publicly available datasets and also compared their performance against popular pretrained LMs. Our results demonstrate that Watson Assistant outperforms mar-

[7] https://www.ibm.com/cloud/watson-assistant

[8] https://cloud.google.com/dialogflow

[9] https://www.luis.ai

[10] https://rasa.com

ket competitors on the HINT3 dataset collection, which comprises real-world queries. Our results also show that Watson Assistant is competitive with pretrained LMs across a wide range of datasets and settings but trains much faster - which is a key factor in usability of a commercial conversational AI solution.

References

Gaurav Arora, Chirag Jain, Manas Chaturvedi, and Krupal Modi. 2020. HINT3: Raising the bar for intent detection in the wild. In *Proceedings of the First Workshop on Insights from Negative Results in NLP*, Online. Association for Computational Linguistics.

Daniel Braun, Adrian Hernandez Mendez, Florian Matthes, and Manfred Langen. 2017. Evaluating natural language understanding services for conversational question answering systems. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 174–185, Saarbrücken, Germany. Association for Computational Linguistics.

Tanja Bunk, Daksh Varshneya, Vladimir Vlasov, and Alan Nichol. 2020. Diet: Lightweight language understanding for dialogue systems.

Iñigo Casanueva, Tadas Temcinas, Daniela Gerz, Matthew Henderson, and Ivan Vulic. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Seattle.

Daniel Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 20th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4171–4186, Minneapolis. The Association for Computational Linguistics.

Stefan Larson, Anish Mahendran, Joseph Peper, Christopher Clarke, Andrew Lee, P. Hill, Jonathan K. Kummerfeld, Kevin Leach, M. Laurenzano, L. Tang, and J. Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *EMNLP/IJCNLP*.

X. Liu, A. Eshghi, P. Swietojanski, and Verena Rieser. 2019a. Benchmarking natural language understanding services for building conversational agents. *ArXiv*, abs/1903.05566.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, pages 1–13.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 19th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL)*, pages 2227–2237, New Orleans. The Association for Computational Linguistics.

Patti Price. 1990. Evaluation of spoken language systems: The atis domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008, Long Beach, CA.

Shi Yu, Yuxin Chen, and Hussain Zaidi. 2020. A financial service chatbot based on deep bidirectional transformers.