SUITABILITY OF DEEP LEARNING ARCHITECTURES FOR FACIAL RECOGNITION

Betzalel Moskowitz
MSDS458-DL: Artificial Intelligence and Deep Learning
July 23, 2023

Abstract

This paper discusses important modeling considerations when developing intent recognition systems for conversational agents. To highlight some of these important modeling considerations, the AG news dataset was used as a proxy problem to perform a classification task on news articles to predict the topic category. The experiments used a variety of different model architectures, representations, and vocabulary sizes to demonstrate the power of recurrent-based models for classification tasks and the importance of attaining strong feature representations over algorithm development. Amongall model architectures, increasing the vocab size from 1000 to 10,000 led to over a four-percentage point increase in test accuracy, with the best performing model achieving a test accuracy of 90.75%.

Introduction

Conversational agents (chatbots) deliver tremendous business value – they make customer service more efficient and less costly, gauging inquiries from customers and routing them to the resources that are of most help to them. For instance, a chatbot may point a customer to a resource showing how to reset their password or it may direct them to a customer service representative that may be able to better satisfy the customer's needs. To do this, the chatbot must make inferences about the topic of the customer's inquiry to route them to the best possible resource for assistance. While today's most advanced chatbots such as ChatGPT require massive language models with trillions of parameters and reinforcement learning, simpler chatbots geared toward providing business-related assistance can be achieved with limited amounts of data and computational overhead.

To develop an understanding of the text, chatbots require natural language understanding (NLU), or the ability to organize the unstructured nature of the text so the chatbot can make sense of it. NLU is achieved through both syntax and semantic analysis (Dilmegani 2022).

Syntax analysis involves identifying the basic grammar rules and language syntax by first preprocessing the text through stemming/lemmatization and removing stop words that provide little meaningful information. The text is also tokenized and part-of-speech (PoS) tagged using supervised learning algorithms such as SVM, Bayesian networks, and maximum entropy algorithms (Dilmegani 2022).

Semantic analysis involves understanding the meaning of the text through the context of each word as well as the relationships between words. This type of analysis often involves unsupervised algorithms (Dilmegani 2022).

In order to recognize the intent of the customer's query, the business may use a text classifier to predict the probability of the input text belonging to a certain category of inquiries. For example, a classifier may be trained to determine whether the customer's intent is to manage their account, purchase products, or speak with a customer representative (Dilmegani 2022).

Once the intent has been recognized, the chatbot must provide a response – this response can be one that is pre-defined (or rule-based) or it can be generated on the fly using generative models.

Most commercial conversational agents tend to use pre-defined rules as they deliver predictable responses and ensure equal treatment of customers, avoiding the potential risks of generative models producing inappropriate responses (Dilmegani 2022).

Based on the back-and-forth dialogue between the customer and the conversational agent, the agent may ask the customer to clarify their response or provide more information to gain the necessary information needed to perform tasks. Many conversational agents will rely on named-entity-recognition models to extract necessary data from user responses.

As generative and rule-based responses are both out of scope, this paper will focus on important modeling principles for building text classifiers, the systems that enable the agent to understand the goal behind the user's query.

As a proxy problem for identifying the customer service area in which the customer requires assistance, the AG News Subset dataset (Zhang et al., 2015) will be used in an attempt to classify news articles into their topic categories.

Literature Review

Papers with Code, a website that tracks performance from different research papers on machine learning tasks, lists 21 papers written on the AG News classification task, with 20 achieving an error rate below 10%.

The first paper published on the AG News classification task was "Character-level Convolutional Networks for Text Classification" (Zhang et al., 2015). The paper introduced "Char-level CNN", a novel approach to text classification by focusing on character level representations and applying one-dimensional CNN architectures to NLP tasks. The paper achieved a 9.51% error rate and demonstrated the effectiveness of CNNs for capturing local patterns and features characteristic of character sequences in text. Additionally, using character-level features allowed the model to overcome previous models' struggles to handle words that were not in the training data's vocabulary since the model worked directly with characters instead of words. This model performed best on shorter-sequences, but nonetheless demonstrated that character-level information can be effective where traditional word-level methods fall short (Zhang et al., 2015).

Another paper, "Disconnected Recurrent Neural Networks for Text Categorization" proposed a novel approach to text categorization using disconnected recurrent neural networks (DRNNs) (Wang, 2018). Prior work with recurrent neural networks (RNNs) on text classification tasks exhibited suffering from vanishing gradient problems, hindering the ability to capture long-range dependencies in text. To address this problem, Wang introduced disconnected recurrent connections within the network architecture, allowing for more effective modeling of distant relationships between words. The proposed DRNN architecture helped improve text categorization performance by better capturing semantic context in sequences. DRNN achieved a test error rate of 5.53% on the AG News dataset (Wang 2023).

"L Mixed", a new approach proposed by Singh Sachan et al. (2020) involved using a mixed objective function to enhance the performance of LSTM-based models. The researchers used both labeled and unlabeled data to improve the accuracy of text classification tasks. To do this, the researchers introduced a hybrid objective function that combined traditional cross-entropy loss with a novel consistency loss, intended to help the model generalize better to unseen data and improve its performance on the semi-supervised task. Through experimental evaluations, the researchers demonstrated that their approach outperformed traditional LSTM-based methods on various text classification tasks when limited labeled data is available. The researchers achieved an error rate of 4.95% (Singh Sachan et al., 2020).

While substantial research has gone into CNN and RNN-based models, the spectacular success of transformer-based models has pushed the benchmark even further. Sun et al. published a study in 2019 demonstrating the effectiveness of fine-tuning pre-trained BERT models for text classification tasks, achieving an error rate of 4.8% on the AG News dataset (2019).

Another 2019 study by Yang et al. proposed XLNet to improve performance further. In order to avoid pretrain-fine-tune discrepancies that are prevalent in BERT, XLNet leverages permutations of words in a sentence to capture bidirectional context and enhance the ability to model dependencies between all words. Unlike previous models like BERT, XLNet treated all words as potential predictions, leading to improved handling of relationships within sentences (Yang et al., 2019). XLNet is currently listed as the state-of-the-art model on Papers with Code for the AG News dataset, achieving an error rate of 4.45% (Papers with Code).

Methods

Data Exploration

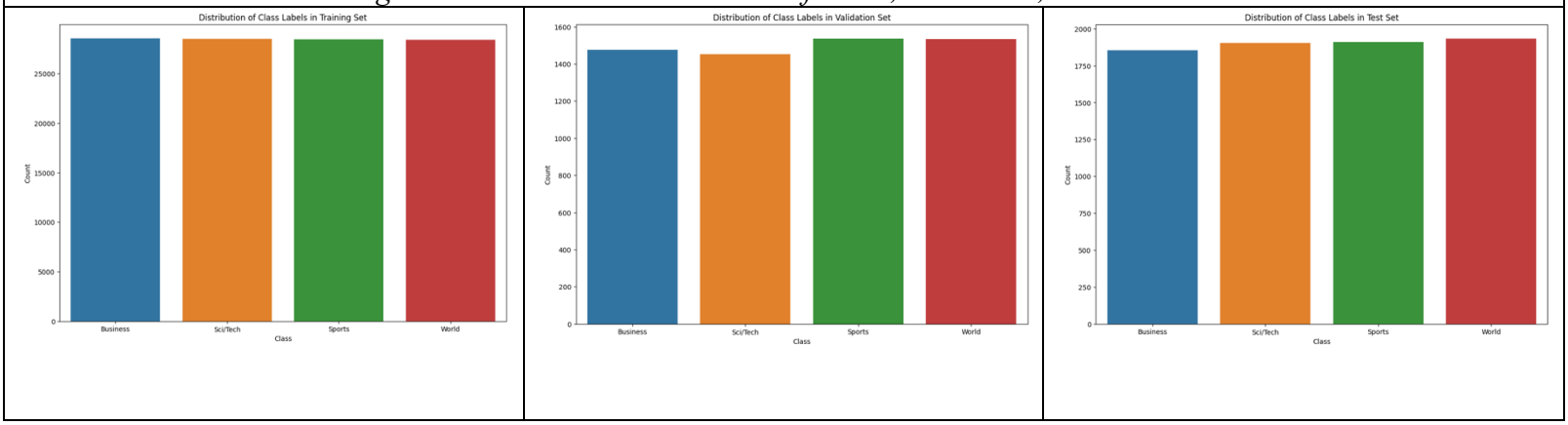
The AG corpus consists of over a million news articles originating from over 2000 news sources (Gulli). The AG news dataset used in this paper was constructed by Xiang Zhang et al. from a subset of the corpus – it contains 127,600 samples (Zhang et al., 2015). As displayed in *Figure 1*, each sample contains the text from the article as well as a numerical label representing one of four news topic categories: "world" (0), "sports" (1), "business" (2), and "Science/Tech" (3).

Figure 1. Example Rows of AG News Dataset

	description	label
0	AMD #39;s new dual-core Opteron chip is designed mainly for corporate computing applications, including databases, Web services, and financial transactions.	3 (Sci/Tech)
1	Reuters - Major League Baseball\Monday announced a decision on the appeal filed by Chicago Cubs\pitcher Kerry Wood regarding a suspension stemming from an\incident earlier this season.	1 (Sports)
2	President Bush #39;s quot;revenue-neutral quot; tax reform needs losers to balance its winners, and people claiming the federal deduction for state and local taxes may be in administration planners #39; sights, news reports say.	2 (Business)
3	Britain will run out of leading scientists unless science education is improved, says Professor Colin Pillinger.	3 (Sci/Tech)
4	London, England (Sports Network) - England midfielder Steven Gerrard injured his groin late in Thursday #39;s training session, but is hopeful he will be ready for Saturday #39;s World Cup qualifier against Austria.	1 (Sports)
5	TOKYO - Sony Corp. is banking on the \$3 billion deal to acquire Hollywood studio Metro-Goldwyn-Mayer Inc...	0 (World)
6	Giant pandas may well prefer bamboo to laptops, but wireless technology is helping researchers in China in their efforts to protect the engandered animals living in the remote Wolong Nature Reserve.	3 (Sci/Tech)
7	VILNIUS, Lithuania - Lithuania #39;s main parties formed an alliance to try to keep a Russian-born tycoon and his populist promises out of the government in Sunday #39;s second round of parliamentary elections in this Baltic country.	0 (World)
8	Witnesses in the trial of a US soldier charged with abusing prisoners at Abu Ghraib have told the court that the CIA sometimes directed abuse and orders were received from military command to toughen interrogations.	0 (World)
9	Dan Olsen of Ponte Vedra Beach, Fla., shot a 7-under 65 Thursday to take a one-shot lead after two rounds of the PGA Tour qualifying tournament.	1 (Sports)

The entire dataset is comprised of 31,900 samples belonging to each of the four datasets. 7,600 samples were set aside for the test set, while five percent of the remaining training dataset was set aside for validation. *Figure 2* shows that each of the datasets contain a roughly uniform distribution of classes, rendering the datasets well-balanced. This makes test-accuracy a good metric for quantifying performance.

Figure 2. Class Label Distribution of Train, Validation, and Test Sets

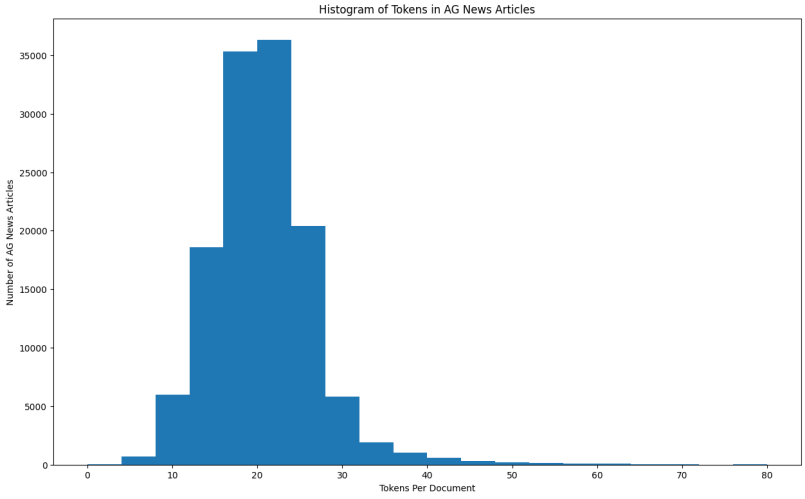


Following the removal of stop words, there were 95,287 unique vocabulary words in the corpus. Descriptive statistics of the number of words per article are displayed in *Figure 3*. *Figure 4* provides a histogram to visualize the distribution of words per article.

Figure 3: Descriptive Statistics of Words Per Article

Mean	20.0
Median	20.21
Standard Deviation	6.21
Minimum	2
Maximum	95

Figure 4: Histogram of Words Per Article



As seen in *Figure 4,* the distribution seems to resemble a normal distribution (with a slight right tail). 40 was chosen as the max length for all input sequences as this value is close to three times the standard deviation plus the mean, where 99.7% of the values should ideally lie.

Data Preprocessing

Upon retrieval, the AG-News dataset was split into train (114,000 samples), validation (6,000 samples), and test (7,600 samples) sets. The text was vectorized with forty tokens, stop words were removed, and the vocabulary size was set based on the experiment (the paper tests vocabulary sizes of 1,000 versus 10,000).

Implementation and Programming

The experiments were carried out via Google Collaboratory notebooks, utilizing Google's T4 GPUs on a hosted Python runtime. All modeling was conducted using Keras within TensorFlow (version 2.12). All visualizations were created using Matplotlib or Seaborn. Additional Python libraries used included the Pandas and NumPy libraries for data manipulation.

An ablation study was conducted in an attempt to answer the following questions:

- How does the vocabulary size affect the performance of the model?
- Do certain model architectures (Fully Connected Layers, RNN, LSTM, GRUs, 1D CNN) perform better than others for text classification tasks?
- What representation techniques yield stronger performance (one-hot encoding vs. dense embeddings)
- Do bidirectional models yield stronger results than unidirectional models?
- How effective are different regularization techniques at reducing overfitting?

To operationalize the experiments, certain hyperparameters remained fixed throughout all experiments. All text sequences were capped at a length of 40. ReLU and RMSProp were chosen as the activation function and optimizer respectively. Since the labels were encoded as integers, sparse categorical cross-entropy was selected for the loss function. Each experiment was allowed to train on up to 200 epochs with a batch size of 128 and with early stopping (patience=3). Early stopping occurred if the validation accuracy did not improve after three epochs, preserving the best model and avoiding additional overfitting.

This study focused on training fully connected dense networks, RNNs, LSTMs, 1D CNNs and GRUs for the text classification text. Models trained were repeated such that each model could be trained with a vocabulary size of 1,000 and 10,000. A version of each model was also trained using both representation techniques – one hot encoding and dense embeddings with output sizes of 256. RNNs, LSTMs, and GRUs were trained with both unidirectional and bidirectional architectures. RNN, LSTM, and GRU layers were trained with 128 hidden units in each layer. Two different architectures of 1D CNNs were trained – one with one layer and 128 filters, and another with two convolutional layers of 256 filters followed by a dense layer of 128 hidden units. Both 1D CNN architectures used a kernel size of three and 1D max pooling size of two. The 1D CNNs were regularized with a 50% dropout.

In order to avoid exploding gradients, the hyperbolic tangent (tanh) activation function was used solely for the deep fully connected dense models which were trained with four layers of 256 neurons. All other models ended with a fully connected layer with 64 hidden units and an output layer with 4 hidden units and a SoftMax activation function.

In addition to the models mentioned above, regularization via dropout was also applied to the GRU models, which had shown signs of heavy overfitting when trained without regularization.

Figure 5 contains a breakdown of all experiments conducted. For each of the experiments run, performance metrics, and a confusion matrix were recorded to examine training and performance.

Figure 5: Description of Experiments

Model	Dense Layers Activation Function	Representation	Vocab Size	Max Length	Uni/Bidirectional	Trainable Params	Regularization
DNN - 4 layers of 256 Neurons	tanH	One-hot	1000	40	Unidirectional	208,900	50% Dropout
RNN (128 Units)	ReLU	One-hot	1000	40	Unidirectional	153,028	None
RNN (128 Units)	ReLU	One-hot	1000	40	Bidirectional	305,732	None
LSTM (128 Units)	ReLU	One-hot	1000	40	Unidirectional	586,564	None
LSTM (128 Units)	ReLU	One-hot	1000	40	Bidirectional	1,172,804	None
GRU (128 Units)	ReLU	One-hot	1000	40	Unidirectional	442,436	None
GRU (128 Units)	ReLU	One-hot	1000	40	Bidirectional	884,548	None
GRU (128 Units)	ReLU	One-hot	1000	40	Bidirectional	884,548	50% Dropout
GRU (128 Units)	ReLU	One-hot	1000	40	Unidirectional	442,436	20% Dropout
GRU (128 Units)	ReLU	One-hot	1000	40	Unidirectional	209,092	25% Recurrent Dropout, 20% Dropout
1D Conv (128 Filters)	ReLU	One-hot	1000	40	Unidirectional	392,644	50% Dropout
RNN (128 Units)	ReLU	Dense Embeddings	1000	40	Unidirectional	313,796	None
RNN (128 Units)	ReLU	Dense Embeddings	1000	40	Bidirectional	371,268	None
LSTM (128 Units)	ReLU	Dense Embeddings	1000	40	Unidirectional	461,636	None
LSTM (128 Units)	ReLU	Dense Embeddings	1000	40	Bidirectional	666,948	None
GRU (128 Units)	ReLU	Dense Embeddings	1000	40	Unidirectional	412,740	None
GRU (128 Units)	ReLU	Dense Embeddings	1000	40	Bidirectional	569,156	None
GRU (128 Units)	ReLU	Dense Embeddings	1000	40	Unidirectional	412,740	20% Dropout
GRU (128 Units)	ReLU	Dense Embeddings	1000	40	Unidirectional	412,740	25% Recurrent Dropout, 20% Dropout
1D Conv (128 Filters)	ReLU	Dense Embeddings	1000	40	Unidirectional	362,948	50% Dropout
1D Conv (256 Filters) 2 Layers	ReLU	Dense Embeddings	1000	40	Unidirectional	691,140	50% Dropout
DNN - 4 layers of 256 Neurons	tanH	One-hot	10000	40	Unidirectional	208,900	50% Dropout
RNN (128 Units)	ReLU	One-hot	10000	40	Unidirectional	1,305,028	None
RNN (128 Units)	ReLU	One-hot	10000	40	Bidirectional	2,609,732	None
LSTM (128 Units)	ReLU	One-hot	10000	40	Unidirectional	5,194,564	None
LSTM (128 Units)	ReLU	One-hot	10000	40	Bidirectional	10,388,804	None
GRU (128 Units)	ReLU	One-hot	10000	40	Unidirectional	3,898,436	None
GRU (128 Units)	ReLU	One-hot	10000	40	Bidirectional	7,796,548	None
GRU (128 Units)	ReLU	One-hot	10000	40	Bidirectional	7,796,548	50% Dropout
GRU (128 Units)	ReLU	One-hot	10000	40	Unidirectional	3,898,436	20% Dropout
GRU (64 Units)	ReLU	One-hot	10000	40	Unidirectional	1,937,092	25% Recurrent Dropout, 20% Dropout
1D Conv (128 Filters)	ReLU	One-hot	10000	40	Unidirectional	3,848,644	50% Dropout
DNN - 4 layers of 256 Neurons	tanH	Dense Embeddings	10000	40	Unidirectional	77,316	50% Dropout
RNN (128 Units)	ReLU	Dense Embeddings	10000	40	Unidirectional	2,617,796	50% Dropout
RNN (128 Units)	ReLU	Dense Embeddings	10000	40	Bidirectional	2,675,268	None
LSTM (128 Units)	ReLU	Dense Embeddings	10000	40	Unidirectional	2,763,636	None
LSTM (128 Units)	ReLU	Dense Embeddings	10000	40	Bidirectional	2,970,948	None
GRU (128 Units)	ReLU	Dense Embeddings	10000	40	Unidirectional	2,716,740	None
GRU (128 Units)	ReLU	Dense Embeddings	10000	40	Bidirectional	2,873,156	None
GRU (128 Units)	ReLU	Dense Embeddings	10000	40	Unidirectional	2,716,740	20% Dropout
1D Conv (128 Filters)	ReLU	Dense Embeddings	10000	40	Unidirectional	2,666,948	50% Dropout
1D Conv (256 Filters) 2 Layers	ReLU	Dense Embeddings	10000	40	Unidirectional	2,995,140	50% Dropout

Results

Vocabulary Sizes

The vocabulary size made a difference across all models – when the vocabulary size was increased from 1,000 to 10,000, the test accuracy among all models gained an extra four to five percent. Excluding the fully connected dense models (whose test accuracy would severely skew the results), the average test accuracies for models using vocabulary sizes 1,000 and 10,000 were 85.83% and 89.98% respectively. The boost in performance was noteworthy – given that there were 95,287 vocabulary words in the corpus, 1,000 tokens was clearly not providing the model with enough features and led to information loss. Limiting the vocabulary size too much may have resulted in many words being treated as out-of-vocabulary (OOV) tokens. By increasing the vocabulary size, more word variations were captured, providing the model with richer information. However, it is worth noting that using all 95,287 vocabulary words may add noise and may not necessarily provide features that are useful - this could harm performance.

Architectures and Representations

Figure 6 displays a chart of the test accuracies of the different network topologies. These averages were taken by averaging the test accuracies of models that share the same topology group despite varying architectures within the topology group. Since there were not sufficient time and resources to train each unique model multiple times to calculate an average test accuracy per model, each of these test accuracies and any conclusions they may suggest should be taken with a grain of salt – with a less than one percent margin, a higher or lower test accuracy may not indicate a model's superiority over another but rather that random initialization of the weights may have given more luck to the superior model in that instance.

Looking at the results data, we would see that the GRU models had the best test accuracy overall. This should be evaluated further, but it's possible that GRUs may have been the best model architecture tested due to their simplified design that renders them less prone to overfitting and therefore may generalize better to unseen data.

One conclusion that can safely be made is that fully connected dense networks do not perform well on their own for NLP tasks – the results show that the models built with this architecture performed as well or barely better than guessing one of the four labels at random. This architecture's ineptitude for this task stems from its inability to model sequential data - since fully connected layers treat input data as flat vectors, they don't inherently understand the sequential relationships between words in a sentence. Texts are inherently sequential data and ignoring this sequential aspect can result in a loss of valuable information. Additionally, fully connected layers don't have mechanisms to capture the context across different positions in the text, a crucial requirement for understanding words and phrases in a sentence.

Figure 6: Average Test Accuracies of Model Topologies

Topology	*Vocabulary Size*			
	1,000		***10,000***	
Fully Connected	One-Hot	0.2710	One-Hot	0.2991
	Embedding	-	Embedding	**0.2500**
RNN	One-Hot	0.8528	One-Hot	0.8957
	Embedding	0.8547	Embedding	0.8974
LSTM	One-Hot	0.8589	One-Hot	0.8980
	Embedding	0.8651	Embedding	0.9027
GRU	One-Hot	0.8610	One-Hot	0.9006
	Embedding	0.8587	Embedding	**0.9031**
1D CNN	One-Hot	0.8555	One-Hot	0.9007
	Embedding	0.8544	Embedding	0.8974

From purely looking at the data, it is difficult to assess whether the one-hot encoding or dense embedding provided better representations, since many of these values are so close to one another and they seem to beat the other in different categories. However, this may be a result of the text classification task not requiring such advanced awareness of context and semantic information.

What can be seen is that the two highest performing model topologies are GRU models with dense embeddings and LSTMs with dense embeddings. The industry has moved away from one-hot encoded representations since embeddings generally provide better semantic information, reduce dimensionality, capture context, and generalize better to unseen or rare words. This can be seen in practice today as most modern models built for text classification fine-tune transformer based models or use transformer models to extract feature embeddings from their dataset before training a classifier.

Unidirectional vs Bidirectional Layers

The result did not suggest that there was a substantial difference between using bidirectional layers and unidirectional layers. This was true for RNNs, LSTMs, and GRUs. While bidirectional layers are generally better at capturing long-range dependencies, disambiguating text, and capturing context, this task may not require bidirectional context to make accurate predictions. Alternatively, since the features were extracted using the same methods and text classification tasks are sensitive to the way that features are extracted and preprocessed, it's possible that the differences between the models may be less pronounced.

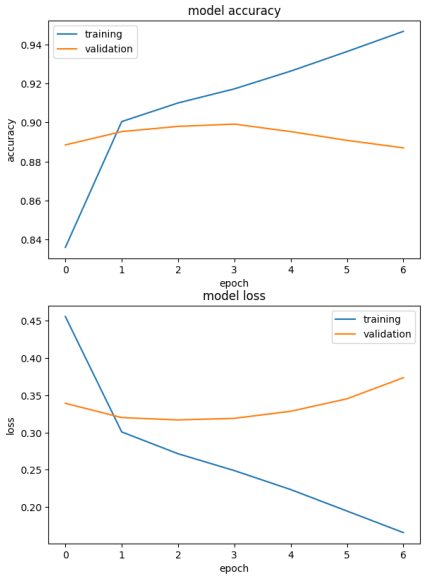
Regularization and Overfitting

Early stopping with a small patience of three was used to mitigate overfitting within the models. For some models, this worked well, and no additional regularization techniques were needed. For

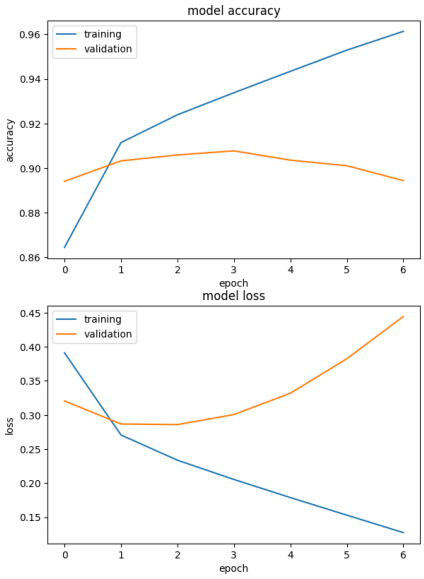
others like the RNNs, GRUs, and 1D convolutional neural networks, overfitting began as early as the first and second epochs. The overfitting can be visualized in *Figure 7*. LSTMs seemed to suffer less from overfitting than the other models – the LSTMs often trained for two to four times more epochs than other topologies, even those with dropout layers.

Figure 7. Overfitting Across Architectures

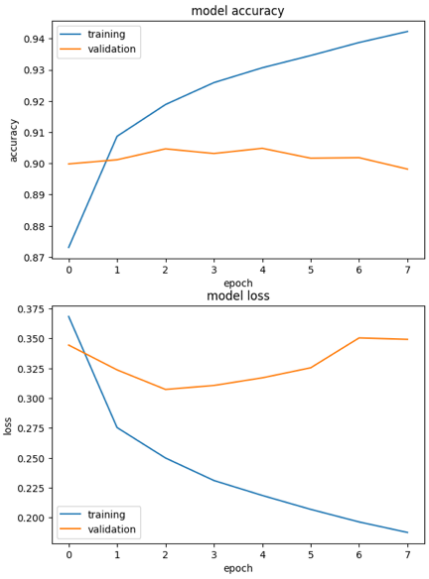
Vocab Size 10,000 Bidirectional RNN with Embedding



Vocab Size 10,000 GRU with Embedding

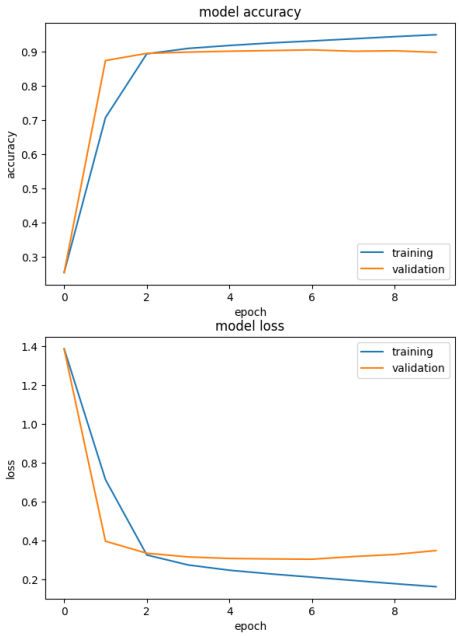
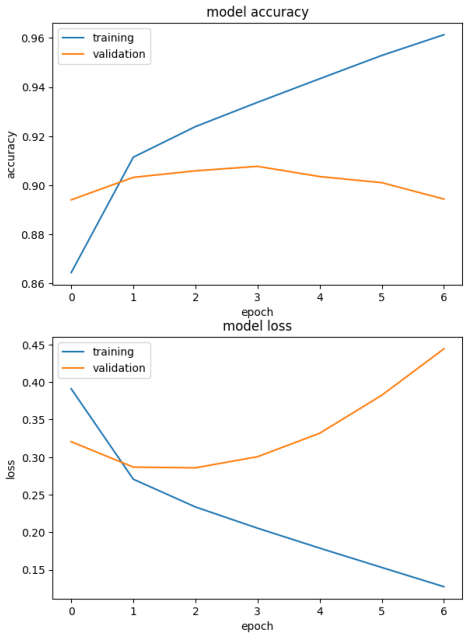


Vocab Size 10,000 1D Convnet with Embedding



While adding dropout layers helped the models to generalize better, the biggest cause of overfitting seemed to come from the representation method used. As can be seen in *Figure 8*, with two GRU models whose architectures are the same other than the representation technique used, the embedding representation tended to overfit earlier and more drastically than the one-hot encoded representation. This trend was prevalent among almost all models indicating that although embedding representations yield high test accuracies, they are more likely to overfit. This is likely because embeddings are typically lower-dimensional representations compared to one-hot encodings. While this can help capture meaningful relationships between data points, it may lead the model to learn the noise in the data. With a lower-dimensional representation, the model has fewer degrees of freedom to generalize complex patterns, which might make it more prone to overfitting. Adequate regularization techniques must be used to ensure that more advanced models generalize well.

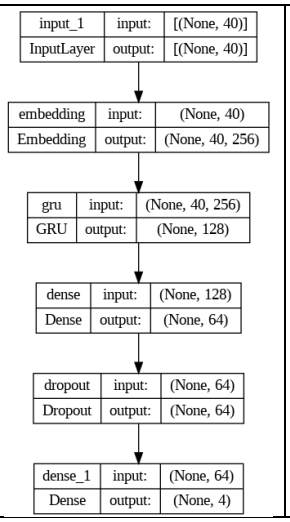
Figure 8. Difference in Overfitting Between

GRU with 20% Dropout, Vocabulary Size 10,000, One-Hot Encoded Representation	GRU with 20% Dropout, Vocabulary Size 10,000, Embedding Representation
	

Best Model

After conducting the experiments, performance metrics were compiled - the full results are available in *Appendix A1*. The best model was a GRU using dense embeddings and a vocabulary size of 10,000, along with a 20% dropout layer between dense layers. The model had 2,716,740 trainable parameters and achieved a 94.38% train accuracy, 90.77% validation accuracy, 90.75% test accuracy, as well as a 91% test precision and recall. This model trained for seven epochs and overfit severely after training for just an epoch or two, but early stopping preserved a better state of the model. The architecture, training visualizations, and confusion matrix are available in *Figures 9, 10, and 11* respectively. The model would benefit from further regularization to reduce overfitting as well as a t-SNE plot to help reduce the dimensionality of the activation layers and visualize the models learned representations.

Figure 9. Architecture of Best Model




```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 40)]              0

embedding (Embedding)        (None, 40, 256)           2560000

gru (GRU)                    (None, 128)               148224

dense (Dense)                (None, 64)                8256

dropout (Dropout)            (None, 64)                0

dense_1 (Dense)              (None, 4)                 260

=================================================================
Total params: 2,716,740
Trainable params: 2,716,740
Non-trainable params: 0
```

Figure 10. Model Training

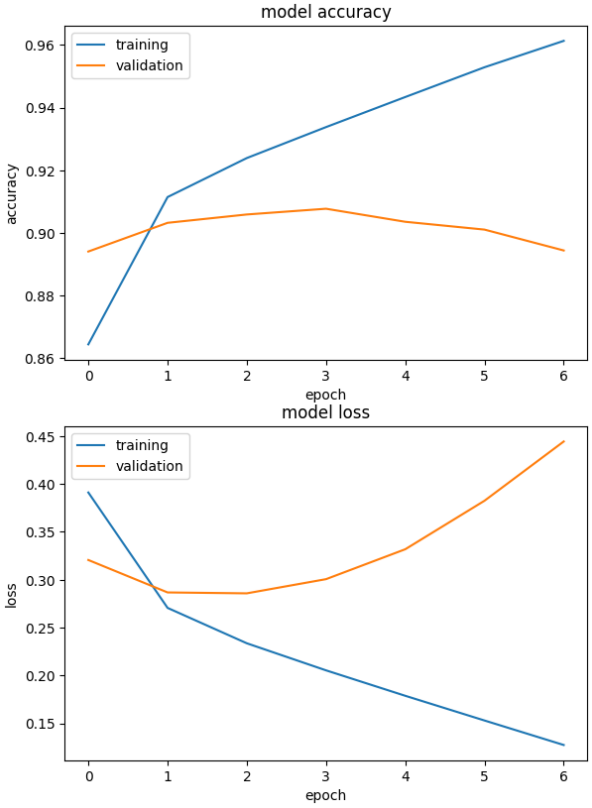
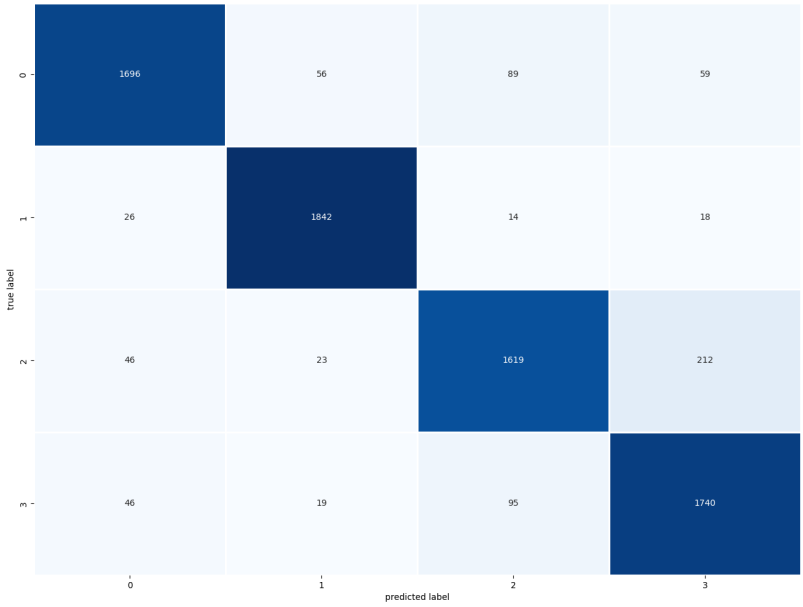


Figure 11. Model Confusion Matrix



Future Work

While a significant amount of time and resources were spent experimenting with different architectures and hyperparameters, the improved performance by increasing the vocabulary size to an appropriate value demonstrates that time might have been better spent exploring and improving the data to provide superior performance. Vocabulary sizes larger than 10,000 should be experimented with to assess whether model performance benefits from additional information. While the model performed well, many models such as transformer-based models may perform even better. Future work might entail working with larger models, fine-tuning a BERT model or using another transformer model to extract better features from the text before stacking a fully connected dense layer on top. Additional work may look at improving generalization with more regularization techniques.

Conclusion

Conversational agents provide business value, especially through more efficient and lower-costing customer service. The success of a conversational agent is determined by the ability for a model to make more accurate inferences on text data inputted by the customer. This ability can be evaluated on text classification tasks, such as the one explored in this study.

While many engineers spend most of their time improving their models through experimentation with different hyperparameters or architectures, the improved performance of changing the vocabulary size in our experiments demonstrates that it is often better to put more time into exploring and preprocessing the data than algorithm development. The improved performance of the sequential-oriented models versus the fully connected dense network demonstrates the

importance of understanding the nature of the available data and the assumptions, strengths, and weaknesses of algorithms should be considered when selecting model architectures.

References

Dilmegani, C. (2022, December 29). *In-depth guide into Chatbots intent recognition in 2023*. In-Depth Guide Into Chatbots Intent Recognition in 2023. https://research.aimultiple.com/chatbot-intent/

Gulli, A. (n.d.). AG's corpus of news articles. http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html

Papers with Code. (n.d.). *Text Classification on AG News | Papers With Code*. Papers with Code: The Latest in Machine Learning. https://paperswithcode.com/sota/text-classification-on-ag-news

Singh Sachan, D., Zaheer, M., & Salakhutdinov, R. (2020). Revisiting LSTM Networks for Semi-Supervised Text Classification via Mixed Objective Function. arXiv e-prints, arXiv-2009.

Sun, C., Qiu, X., Xu, Y., & Huang, X. (2019). How to fine-tune bert for text classification?. In Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18 (pp. 194-206). Springer International Publishing.

Wang, B. (2018, July). Disconnected recurrent neural networks for text categorization. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 2311-2320).

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. Advances in neural information processing systems, 32.

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. Advances in neural information processing systems, 28.

Appendix A1. Experiment Results

Model	Dense Layers Activation Function	Representation	Vocab Size	Max Length	Uni/Bidirectional	Trainable Params	Regularization	Epochs	Training Time (secs)	Training Time/Epoch	Train Accuracy	Validation Accuracy	Test Accuracy	Test Precision	Test Recall
DNN - 4 layers of 256 Neurons	tanH	One-hot	1000	40	Unidirectional	208,900	50% Dropout	4	72	18	0.2639	0.2825	0.2710	0.29	0.27
RNN (128 Units)	ReLU	One-hot	1000	40	Unidirectional	153,028	None	9	1,197	133	0.8616	0.8552	0.8534	0.85	0.85
RNN (128 Units)	ReLU	One-hot	1000	40	Bidirectional	305,732	None	6	1,265	210.83	0.8565	0.8560	0.8522	0.85	0.85
LSTM (128 Units)	ReLU	One-hot	1000	40	Unidirectional	586,564	None	9	319	35.44	0.8556	0.8563	0.8543	0.85	0.85
LSTM (128 Units)	ReLU	One-hot	1000	40	Bidirectional	1,172,804	None	22	885	40.23	0.8799	0.8690	0.8634	0.86	0.86
GRU (128 Units)	ReLU	One-hot	1000	40	Unidirectional	442,436	None	14	412	29.43	0.8868	0.8655	0.8628	0.86	0.86
GRU (128 Units)	ReLU	One-hot	1000	40	Bidirectional	884,548	None	9	336	37.33	0.8835	0.8693	0.8646	0.86	0.86
GRU (128 Units)	ReLU	One-hot	1000	40	Bidirectional	884,548	50% Dropout	11	395	35.91	0.8675	0.8667	0.8584	0.86	0.86
GRU (128 Units)	ReLU	One-hot	1000	40	Unidirectional	442,436	20% Dropout	11	339	30.82	0.8728	0.8682	0.8632	0.86	0.86
GRU (128 Units)	ReLU	One-hot	1000	40	Unidirectional	209,092	25% Recurrent Dropout, 20% Dropout	8	4,056	507	0.8574	0.8645	0.8559	0.86	0.86
1D Conv (128 Filters)	ReLU	One-hot	1000	40	Unidirectional	392,644	50% Dropout	7	155	22.14	0.8567	0.8642	0.8555	0.85	0.86
RNN (128 Units)	ReLU	Dense Embeddings	1000	40	Unidirectional	313,796	None	5	1,408	281.6	0.8520	0.8578	0.8530	0.85	0.85
RNN (128 Units)	ReLU	Dense Embeddings	1000	40	Bidirectional	371,268	None	6	3228	538	0.8633	0.8593	0.8564	0.86	0.86
LSTM (128 Units)	ReLU	Dense Embeddings	1000	40	Unidirectional	461,636	None	22	932	42.36	0.8755	0.8653	0.8649	0.87	0.86
LSTM (128 Units)	ReLU	Dense Embeddings	1000	40	Bidirectional	666,948	None	19	1480	77.89	0.8792	0.8698	0.8653	0.86	0.87
GRU (128 Units)	ReLU	Dense Embeddings	1000	40	Unidirectional	412,740	None	9	386	42.89	0.8580	0.8578	0.8553	0.86	0.86
GRU (128 Units)	ReLU	Dense Embeddings	1000	40	Bidirectional	569,156	None	6	436	72.67	0.8560	0.8585	0.8578	0.86	0.86
GRU (128 Units)	ReLU	Dense Embeddings	1000	40	Unidirectional	412,740	20% Dropout	9	393	43.67	0.8773	0.8708	0.8630	0.86	0.86
GRU (128 Units)	ReLU	Dense Embeddings	1000	40	Unidirectional	412,740	25% Recurrent Dropout, 20% Dropout	4	1564	391	0.8088	0.8413	0.8588	0.86	0.86
1D Conv (128 Filters)	ReLU	Dense Embeddings	1000	40	Unidirectional	362,948	50% Dropout	7	167	23.86	0.8580	0.8637	0.8570	0.86	0.86
1D Conv (256 Filters) 2 Layers	ReLU	Dense Embeddings	1000	40	Unidirectional	691,140	50% Dropout	5	214	42.8	0.8443	0.8580	0.8517	0.85	0.85
DNN - 4 layers of 256 Neurons	tanH	One-hot	10000	40	Unidirectional	208,900	50% Dropout	5	96	19.2	0.2743	0.2912	0.2991	0.33	0.30

RNN (128 Units)	ReLU	One-hot	10000	40	Unidirectional	1,305,028	None	7	850	121.43	0.9172	0.8992	0.8968	0.90	0.90
RNN (128 Units)	ReLU	One-hot	10000	40	Bidirectional	2,609,732	None	6	1449	241.5	0.9154	0.8990	0.8946	0.89	0.89
LSTM (128 Units)	ReLU	One-hot	10000	40	Unidirectional	5,194,564	None	8	497	62.13	0.9057	0.9000	0.8920	0.89	0.89
LSTM (128 Units)	ReLU	One-hot	10000	40	Bidirectional	10,388,804	None	10	1222	122.2	0.9213	0.9070	0.9039	0.90	0.90
GRU (128 Units)	ReLU	One-hot	10000	40	Unidirectional	3,898,436	None	10	650	65	0.9312	0.8982	0.8950	0.89	0.90
GRU (128 Units)	ReLU	One-hot	10000	40	Bidirectional	7,796,548	None	7	744	106.29	0.9353	0.9025	0.9005	0.90	0.90
GRU (128 Units)	ReLU	One-hot	10000	40	Bidirectional	7,796,548	50% Dropout	8	813	101.63	0.9327	0.9002	0.8995	0.90	0.90
GRU (128 Units)	ReLU	One-hot	10000	40	Unidirectional	3,898,436	20% Dropout	10	610	61	0.9322	0.9060	0.9060	0.91	0.91
GRU (64 Units)	ReLU	One-hot	10000	40	Unidirectional	1,937,092	25% Recurrent Dropout, 20% Dropout	7	3687	526.71	0.9182	0.9063	0.9021	0.90	0.90
1D Conv (128 Filters)	ReLU	One-hot	10000	40	Unidirectional	3,848,644	50% Dropout	10	614	61.4	0.9198	0.9018	0.9007	0.90	0.90
DNN - 4 layers of 256 Neurons	tanH	Dense Embeddings	10000	40	Unidirectional	77,316	50% Dropout	7	109	15.57	0.2500	0.2515	0.2500	0.06	0.25
RNN (128 Units)	ReLU	Dense Embeddings	10000	40	Unidirectional	2,617,796	50% Dropout	6	1714	285.67	0.9146	0.9020	0.8970	0.90	0.90
RNN (128 Units)	ReLU	Dense Embeddings	10000	40	Bidirectional	2,675,268	None	6	1832	305.33	0.9164	0.9035	0.8978	0.90	0.90
LSTM (128 Units)	ReLU	Dense Embeddings	10000	40	Unidirectional	2,763,636	None	13	538	41.38	0.9273	0.9058	0.9021	0.90	0.90
LSTM (128 Units)	ReLU	Dense Embeddings	10000	40	Bidirectional	2,970,948	None	14	1022	73	0.9356	0.9063	0.9033	0.90	0.90
GRU (128 Units)	ReLU	Dense Embeddings	10000	40	Unidirectional	2,716,740	None	10	429	42.9	0.9209	0.9053	0.8999	0.90	0.90
GRU (128 Units)	ReLU	Dense Embeddings	10000	40	Bidirectional	2,873,156	None	12	822	68.5	0.9289	0.9055	0.9018	0.90	0.90
GRU (128 Units)	ReLU	Dense Embeddings	10000	40	Unidirectional	2,716,740	20% Dropout	7	306.96	43.85	0.9338	0.9077	**0.9075**	0.91	0.91
1D Conv (128 Filters)	ReLU	Dense Embeddings	10000	40	Unidirectional	2,666,948	50% Dropout	8	219	27.38	0.9306	0.9048	0.9036	0.90	0.90
1D Conv (256 Filters) 2 Layers	ReLU	Dense Embeddings	10000	40	Unidirectional	2,995,140	50% Dropout	4	150	37.5	0.8604	0.8945	0.8912	0.89	0.89