Recurrent Events: A Comparison of Feedforward and Recurrent Neural Networks for Speech Recognition

Tyler Lucher

Department of Computer Science, Binghamton University

CS301: Ethical, Social, and Global Issues in Computing

Dr. George Weinschenk

May 15, 2021

Abstract

Speech recognition algorithms affect the accuracies and speeds of voice assistants, which need to process user inputs to perform essential functions. Speech recognition relies on feedforward neural networks (FFNNs) and recurrent neural networks (RNNs) to compute the probabilities of additional words in a sequence when given the previous word or words. Developers of speech recognition algorithms use trigonal FFNNs and long short-term memory neural networks (LSTMs), the most accurate and fastest respective types of FFNNs and RNNs. In this paper, I compare the accuracies and the computation speeds of FFNNs and RNNs to demonstrate the superiority of RNNs. I compare the accuracies of the neural networks in terms of their word error rates (WERs) and compare speeds of the neural networks in terms of the number of parameters each model needs to access. Martin Sundermeyer, Hermann Ney, and Ralf Schlüter (2015) contribute most significantly to each neural network's model and comparable data. The implementation of RNNs in voice assistants' speech recognition algorithms will increase voice assistants' convenience for users. Disabled people who rely on voice assistant technologies will see additional benefits. Corporations that produce voice assistant technologies may also see more profits from voice assistant products. Computer lawyers need to prevent the corporations and governments' misuse of voice assistant technologies to avoid sacrificing the rights to privacy and free speech. Ethical developers must implement RNNs when considering the utility, common good, and virtue perspectives. In the future, developers should focus on lowering the WERs of speech recognition algorithms for voice assistants.

Recurrent Events: A Comparison of Feedforward and Recurrent Neural Networks for Speech Recognition

Voice assistants have transformed the way people interact with technology over the last 10 years. From Siri to Alexa to Google Home, voice assistants continue to evolve to meet their users' desires. Voice assistant technologies consist of several types of algorithms, including different forms of artificial intelligence, data encryption models, and speech recognition algorithms. Speech recognition algorithms convert their users' input from spoken language into computer-readable text. Voice assistants interpret this converted text to process their users' commands before the assistants develop responses. Algorithms for speech recognition differ in terms of speed and word error rate. Faster speeds and lower word error rates improve speech recognition algorithms' efficiencies and accuracies, respectively. Developers use both feedforward neural networks and recurrent neural networks to implement efficient and accurate speech recognition in voice recognition technologies. Speech recognition algorithm designers implemented feedforward neural networks in the 1980s, and recurrent neural networks in the 2000s. Today, recurrent long short-term memory neural networks (LSTMs) feature prominently in speech recognition technologies. Using recurrent neural networks (RNNs) as opposed to feedforward neural networks (FFNNs) in speech recognition improves voice assistants due to their lower word error rates (WERs). These improvements in speech recognition algorithms aid accessibility, but may complicate issues of privacy and free speech. Recurrent networks improve speech recognition and voice assistants to a better extent than feedforward neural networks.

Precedents and Related Technology

In a journal article for the *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **Martin Sundermeyer, Hermann Ney, and Ralf Schlüter (2015)** detail the history

of neural networks in language modeling. As the authors describe, neural networks create language models to predict the probabilities that sequences of words will appear in sentences for algorithms that rely on patterns, such as speech recognition. Count-based language models that rely on text data predicted these probabilities in the past; however, neural network language models that can predict sequences with a higher number of words outperform count-based language models (Sundermeyer et al., 2015, p. 517). Developers find these features of neural network language models more useful than count-based language models for many tasks, including speech recognition algorithms.

In addition to speech recognition algorithms, the benefits of neural network language models extend to acoustic modeling algorithms. In a separate journal article for the *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, **Abdel-rahman Mohamed, George E. Dahl, and Geoffrey Hinton (2012)** explain that deep belief networks, a type of FFNN with many layers, surpass Gaussian mixture models for the purpose of acoustic modeling. Deep belief networks' advantages include fewer required assumptions about data distribution, easily combined discrete and continuous features, and a higher amount of data that constrains each parameter (Mohamed et al., 2012, p. 14). Acoustic modeling algorithm developers apply FFNNs for these features, though this does not necessarily mean that FFNNs integrate the same improvements in the context of speech recognition. As a result, language modeling processes for speech recognition use LSTMs, RNNs, and FFNNs.

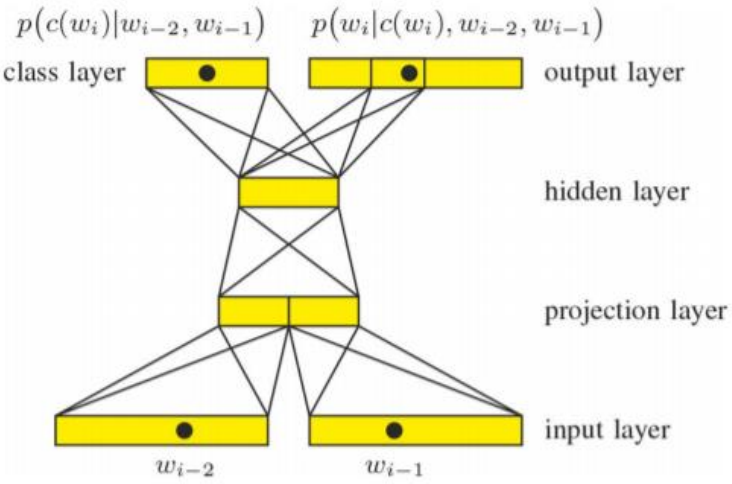
Sundermeyer et al. (2015) provide an overview of how FFNNs function in speech recognition. The authors modeled the Markov assumption with the equation

$$p(w_1^I) = \prod_{i=1}^{I} p\left(w_i \middle| w_{i-n+1}^{i-1}\right) \quad (1)$$

in which the current word, w_i, and the most recent preceding words, w_{i-n+1}^{i-1}, predict the probability of the word sequence w_1^I (Sundermeyer et al., 2015, p. 518). This Markov assumption holds that the previous sequences of the words before the current word do not affect the current word. Figure 1 portrays how Sundermeyer et al. model the architecture of trigram FFNNs due to trigram FFNNs' lower levels of perplexity that allow them to decrease processing time without affecting accuracy (Sundermeyer et al., 2015, pp. 518–519).

Figure 1

The Architecture of a Trigram FFNN



Several equations describe the transformation of w_{i-1} and w_{i-2}, the input word data, to $p(c(w_i)|w_{i-2}, w_{i-1})$ and $p(w_i|c(w_i), w_{i-2}, w_{i-1})$, the probabilities of word sequences in the output layer. In the following equations, weight matrices A_1, A_2, A_3, and A_4 all contain the number of rows equal to the number of inputs from the previous data and the number of columns equal to the number of outputs for the next data. The equation

$$y_i = A_1 \widehat{w}_{i-2} \circ A_1 \widehat{w}_{i-1} \quad (2)$$

activates the projection layer. The equation

$$z_i = \sigma(A_2 y_i) \tag{3}$$

uses (2) as well as

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{4}$$

to activate the hidden layer. The equation

$$p(c(w_i)|w_{i-2}, w_{i-1}) = \varphi(A_3 z_i)|_{C(w_i)} \tag{5}$$

activates the class layer, as seen in the top-left corner of Figure 1, and the equation

$$p(w_i|c(w_i), w_{i-2}, w_{i-1}) = \varphi\left(A_{4,C(w_i)} z_i\right)\Big|_{w_i} \tag{6}$$

activates the output layer, as seen in the top-right corner of Figure 1. Equations (5) and (6) both use the softmax activation function

$$\varphi(x)|_j = \frac{\exp(x_j)}{\sum_{k=1}^{|x|} \exp(x_k)} \tag{7}$$

to normalize the probability estimates. According to Sundermeyer et al., trigonal FFNN language modeling techniques follow the format of equations 1–7 due to low perplexities and WERs, as well as quick computational times (Sundermeyer et al., 2015, pp. 518–519). Speech recognition algorithm developers favor these features of trigonal FFNNs. The accuracies of trigonal FFNNs continue to improve over time as developers discover new techniques.

The processes behind dropout and rectified linear units both benefit FFNNs. According to MIT and University of Toronto researchers **George E. Dahl, Tara N. Sainath, and Geoffrey Hinton (2013)** in an article for the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, both techniques aid Large Vocabulary Continuous Speech Recognition (LVCSR). Speech recognition algorithm developers apply dropout to FFNNs to select which input and hidden layers the algorithm removes randomly and efficiently from each input and hidden layer during training. The developers implement rectified linear units in FFNNs to set

negative results of the activation function to zero. Dahl et al. combine these techniques to examine their synergistic effect on speech recognition neural networks. The researchers test these neural networks with two sets of 50 hours of English Broadcast News. Dahl et al. observe that for both sets of speech data, neural networks with additional dropout and rectified linear unit techniques obtained lower WERs (2013, pp. 8610–8612). While Dahl et al. do not compare the effects of dropout and rectified linear units between different types of neural networks, these techniques may also improve RNNs' accuracies. Because of low WERs and fast computation speeds, speech recognition algorithm developers choose trigonal FFNNs or RNNs to improve voice assistants.

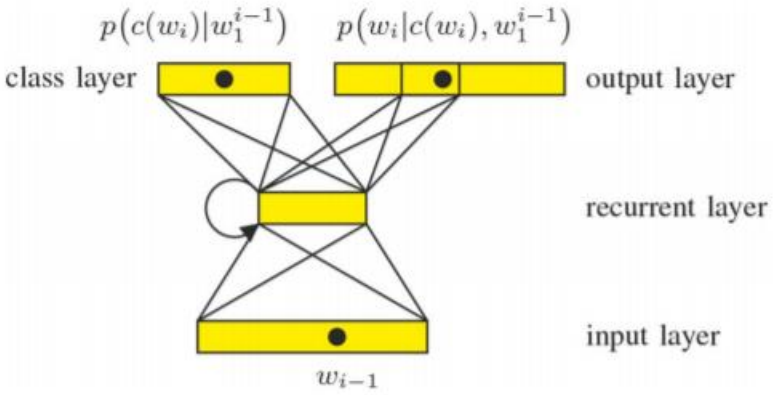
Support

Technical Details

Continuing their analysis, Sundermeyer et al. (2015) find that RNNs outdo the trigonal FFNNs. Figure 2 displays how Sundermeyer et al. model the standard RNN's architecture to demonstrate their similarity to trigonal FFNNs (Sundermeyer et al., 2015, p. 519).

Figure 2

The Architecture of an RNN



The input layer contains only one input word data: w_{i-1}. Equations transform this input data layer into the probabilities of word sequences in the output layer, $p(c(w_i)|w_1^{i-1})$ and $p(w_i|c(w_i), w_1^{i-1})$. In the following equations, weight matrices A_1, A_2, and A_3 contain the number of rows equal to the number of inputs from the previous recursion, and the number of columns equal to the number of outputs for the next recursion. The equation

$$y_i = \sigma(A_1 \widehat{w}_{i-1} + R y_{i-1}) \quad (8)$$

multiplies the previous hidden layer activation vector, y_{i-1}, with a weight parameter matrix, R, to activate the recurrent layer with (4). During the first transformation from the input layer to the recurrent layer, $i = 1$ and $y_{i-1} = 0$, so the equation

$$y_i = \sigma(A_1 \widehat{w}_{i-1}) \quad (9)$$

models this first transformation instead. The equation

$$p(c(w_i)|w_1^{i-1}) = \varphi(A_2 y_i)|_{C(w_i)} \quad (10)$$

activates the class layer, as seen in the top-left corner of Figure 2', and the equation

$$p(w_i|c(w_i), w_1^{i-1}) = \varphi\left(A_{3,C(w_i)} y_i\right)\Big|_{w_i} \quad (11)$$

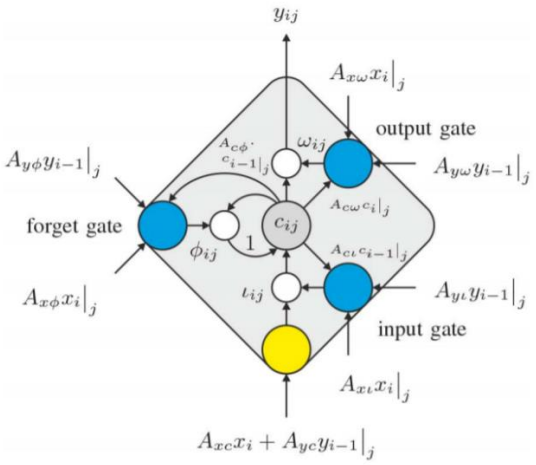
activates the output layer, as seen in the top-right corner of Figure 2. Similar to the output activation functions for trigonal FFNNs, equations (10) and (11) use the softmax activation function (7) to normalize the probability estimates. RNNs use the probability of the previous word sequence, w_1^{i-1}, in the recurrent layer to gradually build a history of words. This word history acts as a memory of word data. Trigonal FFNNs build this history with adjacent words pairs; RNNs instead form the word history with word sequences (Sundermeyer et al., 2015, p. 519). When compared to FFNNs' short-term histories, the independence of RNNs' long-term histories does not require a specified context length for RNNs. Speech recognition algorithms in voice assistants should favor RNNs because voice assistants cannot determine the amount of

speech a user may input. Non-standard RNNs, such as LSTMs, better stabilize the weight parameters when processing a greater number of words.

Before applying equations (10) and (11), LSTMs replace equation (8) with five more equations to replace the recurrent hidden layer with an LSTM layer. Figure 3 illustrates how Sundermeyer et al. model the architecture of the LSTM equations before equations (10) and (11) (Sundermeyer et al., 2015, p. 520).

Figure 3

The Architecture of the LSTM Equations



In the following equations, the variables in the indices of the weight matrices $A_{x\iota}$, $A_{y\iota}$, $A_{c\iota}$, etc., do not represent the dependence of these weight matrices on these variables, but distinguish each weight matrix. The equation

$$\iota_i = \sigma(A_{x\iota}x_i + A_{y\iota}y_{i-1} + A_{c\iota}c_{i-1}) \tag{12}$$

models the input gate, the equation

$$\phi_i = \sigma(A_{x\phi}x_i + A_{y\phi}y_{i-1} + A_{c\phi}c_{i-1}) \tag{13}$$

models the forget gate, and the equation

$$\omega_i = \sigma(A_{x\omega}x_i + A_{y\omega}y_{i-1} + A_{c\omega}c_i) \tag{14}$$

models the output gate. The equations (12), (13), and (14) all result in values in the interval (0, 1). Equations (12), (13), and (14) use (4), as well as the variables x_i, y_i, and c_i. If speech recognition algorithm developers do not require additional projection layers, the variable $x_i = \widehat{w}_{i-1}$; otherwise, the activations of the projection layers map x_i. The equation

$$c_i = \phi_i c_{i-1} + \iota_i \tanh\left(A_{xc} x_i + A_{yc} y_{i-1}\right) \quad (15)$$

uses (13) and (12) to find c_i, and the equation

$$y_i = \omega_i \tanh(c_i) \quad (16)$$

uses (14) to find y_i (Sundermeyer et al., 2015, p. 520). Equations (12) through (16) collectively produce y_i for equations (10) and (11) to use.

In the conference proceedings for the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, **Martin Sundermeyer, Ilya Oparin, Jean-Luc Gauvain, B. Freiberg, Ralf Schlüter, and Hermann Ney (2013)** compare the WERs of FFNNs and LSTMs. Table 1 indicates the researchers' results for singular FFNNs and LSTMs, as well as double-interpolated FFNNs and LSTMS. The researchers use a 100-best list for both FFNNs and LSTMs, and a word lattice for FFNNs. The results for dev12 and test12 represent the results on the Quaero French 2012 development and test acoustic data, respectively (Sundermeyer et al., 2013, p. 8433). The authors decode the data with the Viterbi algorithm and a Confusion Network Combination (CNC) algorithm consisting of speaker adaptation, cross adaptation, Multi-Layer Perceptron, and discriminative training (Sundermeyer et al., 2013, pp. 8431–8432).

Table 1

WER Results of FFNNs and LSTMs

LM Type	Hypoth.	Decoding	WER	
			dev12	test12

+1x FFNN	100-best	Viterbi 1x	14.6%	16.8%
		CNC 5x	14.2%	16.4%
	lattices	Viterbi 1x	15.0%	16.9%
		CNC 5x	14.1%	16.2%
+2x FFNN	100-best	Viterbi 1x	14.6%	16.7%
		CNC 5x	14.1%	16.3%
	lattices	Viterbi 1x	14.9%	16.9%
		CNC 5x	14.1%	16.2%
+1x LSTM	100-best	Viterbi 1x	14.4%	16.3%
		CNC 5x	13.9%	16.0%
+2x LSTM	100-best	Viterbi 1x	14.2%	16.1%
		CNC 5x	13.8%	15.8%

These results demonstrate that the best possible WERs for FFNNs, 14.1% for dev12 and 16.2% for test12, exceed the best possible WERs for LSTMs, 13.8% for dev12 and 15.8% for test12. Additional studies validate LSTMs' superior accuracies.

In another study for the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, **Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton (2013)** corroborate the results of Sundermeyer et al. (2013). The three researchers for the University of Toronto investigate the accuracies of LSTMs for speech recognition. Graves et al. train multiple end-to-end LSTMs with varying numbers of hidden layers on the TIMIT corpus (2013, p. 6648). According to Linguistic Data Consortium members Nattanun Chanchaochai, Christopher Cieri, Japhet Debrah, Hongwei Ding, Yue Jiang, Sishi Liao, Mark Liberman, Jonathan Wright, Jiahong Yuan, Juhong Zhan, and Yuqing Zha (2018), many researchers test the TIMIT dataset on speech

recognition algorithms due to the recording's wide range of speakers (Chanchaochai et al., pp. 192–193). Speech recognition researchers compare between speech recognition algorithms' lowest WERs because of the TIMIT corpus' long-term pervasiveness in the field. Graves et al. find that the end-to-end LSTM speech recognition algorithm achieves an error rate of 17.7%, the least WER at the researchers' paper's time of publication (2013, p. 6648). While Graves et al. never directly compare the WERs between LSTMs and FFNNs, the researchers' results further portray LSTMs' superior accuracies.

The research of Sundermeyer et al. (2015) further supports LSTMs' excellence over FFNNs. Table 2 highlights the researchers' results for singular FFNNs, RNNs, and LSTMs, as well as double-interpolated FFNNs and LSTMs. The researchers vary the number of hidden layers for each neural network and list the results in terms of the perplexity (PPL), character error rate (CER), and WER of each neural network. The researchers also find that the PPLs and CERs both positively correlate to the WERs of neural networks (Sundermeyer et al., pp. 526–527). Lower values for neural networks' PPLs, CERs, and WERs all signify higher accuracies. The lowest PPL, CER, and WER of each neural network appears in bold.

Table 2

PPL, CER, and WER Results of FFNNs, RRNs, and LSTMs

Neural Network	Hidden Layers	PPL	CER	WER
+1x FFNN	100	121.1	7.5	11.8
	200	116.6	7.3	11.6
	300	114.7	7.3	**11.5**
	400	114.1	**7.2**	**11.5**
	500	113.4	**7.2**	**11.5**

	600	**112.5**	**7.2**	**11.5**
+2x FFNN	100	121.2	7.5	11.9
	200	115.7	7.3	11.5
	300	115.4	7.2	11.5
	400	112.2	**7.1**	**11.3**
	500	111.0	**7.1**	**11.3**
	600	**110.2**	7.2	**11.3**
+1x RNN	100	121.0	7.5	11.8
	200	117.6	7.3	11.7
	300	112.6	7.3	11.4
	400	111.5	7.2	11.3
	500	108.9	7.1	11.2
	600	**108.1**	**7.0**	**11.1**
+1x LSTM	100	115.3	7.3	11.7
	200	106.8	7.1	11.2
	300	102.4	6.9	11.0
	400	99.9	6.9	10.9
	500	97.9	6.9	10.9
	600	**96.7**	**6.8**	**10.8**
+2x LSTM	100	111.0	7.2	11.4
	200	101.6	7.0	11.0
	300	97.5	6.8	10.8
	400	95.2	6.9	10.8

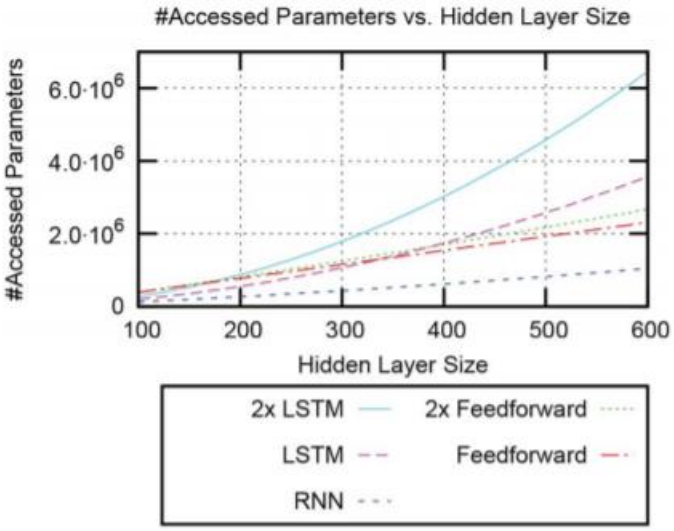
	500	93.1	**6.6**	10.5
	600	**92.0**	6.7	**10.4**

As shown in Table 2, the lowest PPLs, CERs, and WERs of singular FFNNs, double-interpolated FFNNs, singular RNNs, singular LSTMs, and double-interpolated LSTMs decrease, respectively. These results confirm the 2013 Sundermeyer et al. study: the accuracy of LSTMs exceeds the accuracy of FFNNs. The 2015 study introduces that the accuracy of standard RNNs falls between the accuracies of LSTMs and FFNNs. Developers of speech recognition algorithms should prefer a greater number of hidden layers, which increase the accuracy of every model of neural network.

The accuracies of the neural networks do not necessarily affect the neural networks' speeds. Instead, Sundermeyer et al. (2015) found that high numbers of parameters relate to higher speeds of neural networks. Figure 4 models the number of parameters of varying hidden layer sizes of singular FFNNs, RNNs, and LSTMs, as well as double-interpolated FFNNs and LSTMs (Sundermeyer et al., 2015, p. 526).

Figure 4

Accessed Parameters vs. Hidden Layer Sizes of Neural Networks



For higher hidden layer sizes, the number of accessed parameters increases between singular RNNs, singular FFNNs, double-interpolated FFNNs, singular LSTMs, and double-interpolated LSTMs, respectively. FFNNs outperform standard RRNs, but not RRNs that use the LSTM model. Developers should use RNNs for speech recognition algorithms because of their superior accuracies and speeds when compared to FFNNs.

Social Impact

Due to faster computational speeds and lower WERs of RNNs, improvements in the voice assistants' speech recognition algorithms offer benefits to both the corporations that produce voice assistants, and the consumers who purchase products that contain voice assistants. **Logan Kugler (2019)**, a freelance technology and business writer whose work appears in more than 60 major publications, discusses the implications of the ubiquity of facial and voice recognition technologies in an article for the *Communications of the ACM* magazine. He discusses the normalization of consumers' daily use of speech recognition technologies in devices that feature voice assistants, such as Apple's Siri, Amazon's Echo, and Google's Google Home (Kugler, 2019, pp. 518–519). Kugler notes that speech recognition in voice assistants

allows people to ask questions out loud and receive answers in real-time. The ability to search online without needing to type out every inquiry improves the lives of average people who need quick answers to their questions (Kugler, 2019, p. 519). The use of RNNs in the speech recognition algorithms of voice assistants will allow people to search faster and more accurately due to RNNs' decreased computation speeds and WERs. Speech algorithm technologies that implement these advancements also enhance the lives of disabled people.

Disabled people especially benefit from improvements in voice assistants. Logan Kugler (2019) quotes the executive director of the Global Digital Policy Incubator at Stanford University, Eileen Donahoe, to describe how voice recognition improves the accessibility of many technologies for visually impaired people (Kugler, 2019, p. 519). Speech recognition technologies can also improve technological accessibility for people with other disabilities as well. **Neil Savage (2019)**, a freelance science and technology writer who wrote over 600 published works, explains in an article for the *Communications of the ACM* magazine how Tavis Rudd, a self-employed programmer, used Dragon Naturally Speaking voice recognition software to code with numb fingers from a repetitive strain injury (Savage, 2019, pp. 18–19). Tavis Rudd's success demonstrates how voice assistants help people with motor function disabilities complete the same tasks as people without disabilities. The use of RNNs in speech recognition algorithms will improve voice assistants' speeds and accuracies, increasing the accessibility of technology for disabled people.

Despite the benefits resulting from the use of RNNs in voice assistants, the improvements in speech recognition algorithms also create negative social impacts. In a **TED (2018)** Talk, Kashmir Hill and Surya Mattu, two technology reporters for Gizmodo who focus on privacy and security, analyzed the data 18 Internet-connected devices sent to their manufacturers. The

reporters found that the online devices sent an excess of information, including exact schedules for sleeping and brushing teeth (TED). Companies can then apply their users' information to create more targeted products or sell their users' information to third parties. Hackers may take advantage of insecure Internet-connected technologies to leak the sensitive information of at-risk consumers. Potential privacy breaches' implications extend to speech recognition technologies that often use online servers to save time when processing speech. As Hill and Mattu reported, although Hill did not frequently activate her Amazon Echo's Alexa, the voice assistant contacted Internet servers every three minutes (TED). Companies can use speech recognition technologies that detect every word to collect more data than other Internet-connected devices. Companies and third parties that want to target consumers with advertisements value this speech data because speech data provides a high amount of detailed user information.

If developers use RNNs in speech recognition technologies, the cost-effectiveness of this speech data will increase due to faster processing and higher accuracy. As Hill and Mattu mentioned, Internet-connected speech recognition technologies' users may not know that companies collect their information (TED). Unaware consumers may not exercise caution when sharing data with companies; therefore, these companies that sell speech recognition technologies should educate their customers about the privacy risks of their products. Improvements in speech recognition technologies because of RNNs warrant an understandable warning from corporations about how the high amount and quality of the data companies collect increases the risk of a privacy breach.

The government's use of speech technologies improved with RNNs may also sacrifice the privacies of citizens. According to Kugler (2019), due to the ubiquity of facial and speech recognition technologies, governments may easily track their citizens' locations and

conversations. If governments fail to maintain their citizens' privacies, facial and speech recognition technologies will create a chilling effect on the freedoms of assembly and speech (Kugler, 2019, p. 519). If RNNs improve the speeds and accuracies of speech recognition algorithms, governments may easily arrest citizens who say suspicious phrases in specific places. For example, law enforcement can surveille speech recognition technologies to identify someone who says the word "bomb" in an airport. While the government may use speech recognition technology to arrest people who might otherwise commit serious crimes, the WERs of speech recognition technologies never equal zero, and so a false positive rate would always exist. As a result, governments should not use speech recognition technologies to monitor their citizens. Computing professionals should ensure that governments do not purchase speech recognition products for this purpose. Speech recognition algorithm developers must also consider the ethics of the decision to implement RNNs rather than FFNNs.

Ethical Analysis

Speech recognition algorithm developers must examine many ethical viewpoints to verify the decision to use RNNs instead of FFNNs. Markkula Center for Applied Ethics Executive Director Kirk Hanson's (2016) Ethical Decision Making application considers different ethicists' perspectives. John Stuart Mill influences the first concept of utility, which examines whether the decision benefits the most people. In terms of usefulness, RNNs rank higher than FFNNs due to RNNs' higher accuracies and faster processing speeds. Speech recognition algorithm developers who implement RNNs choose a more ethical decision with respect to utility.

Immanuel Kant's rights approach ranks the decisions by how well each choice respects the stakeholders' rights. In this scenario, the stakeholders include the people who would use speech recognition technologies, the developers who improve speech recognition algorithms and

create speech recognition technologies, and corporations that sell these technologies. The decision to favor RNNs or FFNNs does not directly impact the rights of any of these categories. This choice does not affect consumers' access to speech recognition technologies, developers' creation of speech recognition algorithms, and companies' profits from speech recognition technologies. As discussed in the prior section, governments that use speech recognition to police may chill the citizens' rights to free assembly and speech; however, the decision to use RNNs or FFNNs does not affect any governments' choice. Speech recognition algorithm developers who implement both types of neural network respect the rights perspective.

Plato and John Rawls' justice perspective judges whether the decisions treat every individual equally. The choice to use RNNs or FFNNs does not affect one person more than another; everyone's access to speech recognition technologies improves. No matter which neural network the developers implement, no one treats another person as a means to an end. The developers can ethically implement either neural network from the perspective of justice.

John Rawls' common good approach evaluates whether each decision serves the entire community. This perspective must consider the interests of the least-advantaged people. In this situation, the least-advantaged people are those with disabilities. As previously discussed, speech recognition technologies with RNNs improve the lives of disabled people to a greater extent than speech recognition technologies with FFNNs. The developers should apply RNNs to recognize the common good.

Aristotle's virtue perspective tests whether each choice exhibits inherently good character traits. Of Aristotle's list of virtues, only ambition applies to the developers' decision to implement a neural network. The developers who implement RNNs display more ambition through the desire to use a more complicated type of neural network than the developers who

apply FFNNs. Ethical developers should implement RNNs to acknowledge the perspective of virtue.

James Moor's just consequentialism compares ethical perspectives. Table 3 quantifies and weighs the results of the analysis of the Ethical Decision Making app's perspectives (Hanson, 2016).

Table 3

Notes From Ethical Decision Making

Criteria		Decision			
		RNNs		FFNNs	
Ethical Perspective	Weight	Value (out of 100)	Weighted Value	Value (out of 100)	Weighted Value
Utility	.24	100	24.0	80	19.2
Rights	.24	100	24.0	100	24.0
Justice	.24	100	24.0	100	24.0
Common Good	.24	100	24.0	80	19.2
Virtue	.04	100	4.0	80	3.2
Total	1.00	500	100.0	440	89.6

Virtue weighs less than the other perspectives in Table 3 because this viewpoint does not consider the effects of either decision. While the decisions to implement RNNs or FFNNs score equally in the rights and justice categories, developers who choose RNNs over FFNNs select a more ethical decision in terms of utility, common good, and virtue. Ethical speech recognition algorithm developers should use RNNs rather than FFNNs due to RNNs' superiority in these ethical areas.

Conclusion

Developers of speech recognition algorithms for voice assistants must consider the ethics involved with the decision to implement FFNNs or RNNs. The equations used for trigonal FFNNs require a specified context length because the FFNNs' inputs' format necessitates an approximation of the probabilities of the next word in the sequence. The accuracy of RNNs outdoes the accuracy of trigonal FFNNs because the RNNs' recurrent layers construct a longer history of all the word sequences that precede the word that each layer predicts. LSTMs, specific types of RNNs, use an input gate, a forget gate, and an output gate that allow them to better stabilize the weight parameters when processing greater numbers of words. The accuracy of LSTMs outdoes standard RNNs due to this extra stabilization, although beyond the activation of the input layer to the first recurrent layer, both neural networks use identical equations. The computation speeds of LSTMs exceed trigonal FFNNs, which exceed standard RNNs. The number of accessed parameters determines the computation speeds of each neural network. These computation speeds do not necessarily correlate to the accuracy of each neural network. Developers of voice assistants must implement RNNs, as opposed to FFNNs, due to the RNNs' lower WERs and faster speeds. If developers implement RNNs in voice assistants' speech recognition algorithms, people and corporations will benefit. The improved accuracy and speed of voice assistants will satisfy average people who use voice assistants for convenient answers to questions. Disabled people, such as visually impaired people and people with motor function disabilities, who may need voice assistants to complete certain tasks, will benefit the most from improvements in accuracy and speed. The number of people who use voice assistants may increase due to this increased convenience, which will produce more profits for the corporations that produce voice assistant technologies. These benefits may arrive at the cost of people's freedoms. Corporations and governments may use speech recognition algorithms to undermine

the rights to privacy and free speech through constant surveillance. Computer professionals who practice law should pass laws to protect these rights as developers make technological advancements. Ethical developers should favor RNNs due to their superior scores according to the perspectives of utility, common good, and virtue. In the future, the developers should continue to improve the accuracies of neural networks for speech recognition. Speech recognition algorithm developers must use LSTM RNNs in speech recognition algorithms for voice assistants due to low WERs and processing speeds.

References

Chanchaochai, N., Cieri, C., Debrah, J., Ding, H., Jiang, Y., Liao, S., Liberman, M., Wright, J., Yuan, J., Zhan, J., and Zha, J. (2018). *GlobalTIMIT: Acoustic-Phonetic Datasets for the World's Languages*. Proc. Interspeech 2018, 192–196, Hyderabad, India. http://dx.doi.org/10.21437/Interspeech.2018-1185

Dahl, G. E., Sainath, T. N., & Hinton, G. (2013). *Improving deep neural networks for LVCSR using rectified linear units and dropout*. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 8609–8613, Vancouver, Canada. https://doi.org/10.1109/icassp.2013.6639346

Graves, A., Mohamed, A.-R., & Hinton, G. (2013). *Speech recognition with deep recurrent neural networks*. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 6645–6649, Vancouver, Canada. https://doi.org/10.1109/icassp.2013.6638947

Hanson, K. (2016). *Ethical Decision Making* (Version 2.4) [Mobile App]. Apple Store. https://apps.apple.com/us/app/ethical-decision-making/id799710217#?platform=iphone

Kugler, L. (2019, February). Being recognized everywhere. *Communications of the ACM, 62*(2), 17–19. https://doi.org/10.1145/3297803

Mohamed, A.-R., Dahl, G. E., & Hinton, G. (2012). Acoustic modeling using deep belief networks. *IEEE Transactions on Audio, Speech, and Language Processing*, *20*(1), 14–22. https://doi.org/10.1109/tasl.2011.2109382

Savage, N. (2019, May). Code talkers. *Communications of the ACM, 62*(5), 18–19. https://doi.org/10.1145/3317681

Sundermeyer, M., Ney, H., & Schlüter, R. (2015). From feedforward to recurrent LSTM neural networks for language modeling. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *23*(3), 517–529. https://doi.org/10.1109/taslp.2015.2400218

Sundermeyer, M., Oparin, I., Gauvain, J.-L., Freiberg, B., Schlüter, R., & Ney, H. (2013). *Comparison of feedforward and recurrent neural network language models*. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 8430–8434, Vancouver, Canada. https://doi.org/10.1109/icassp.2013.6639310

TED. (2018, August 14). *What your smart devices know (and share) about you | Kashmir Hill and Surya Mattu* [Video]. YouTube. https://www.youtube.com/watch?v=POHYyP4EbzE