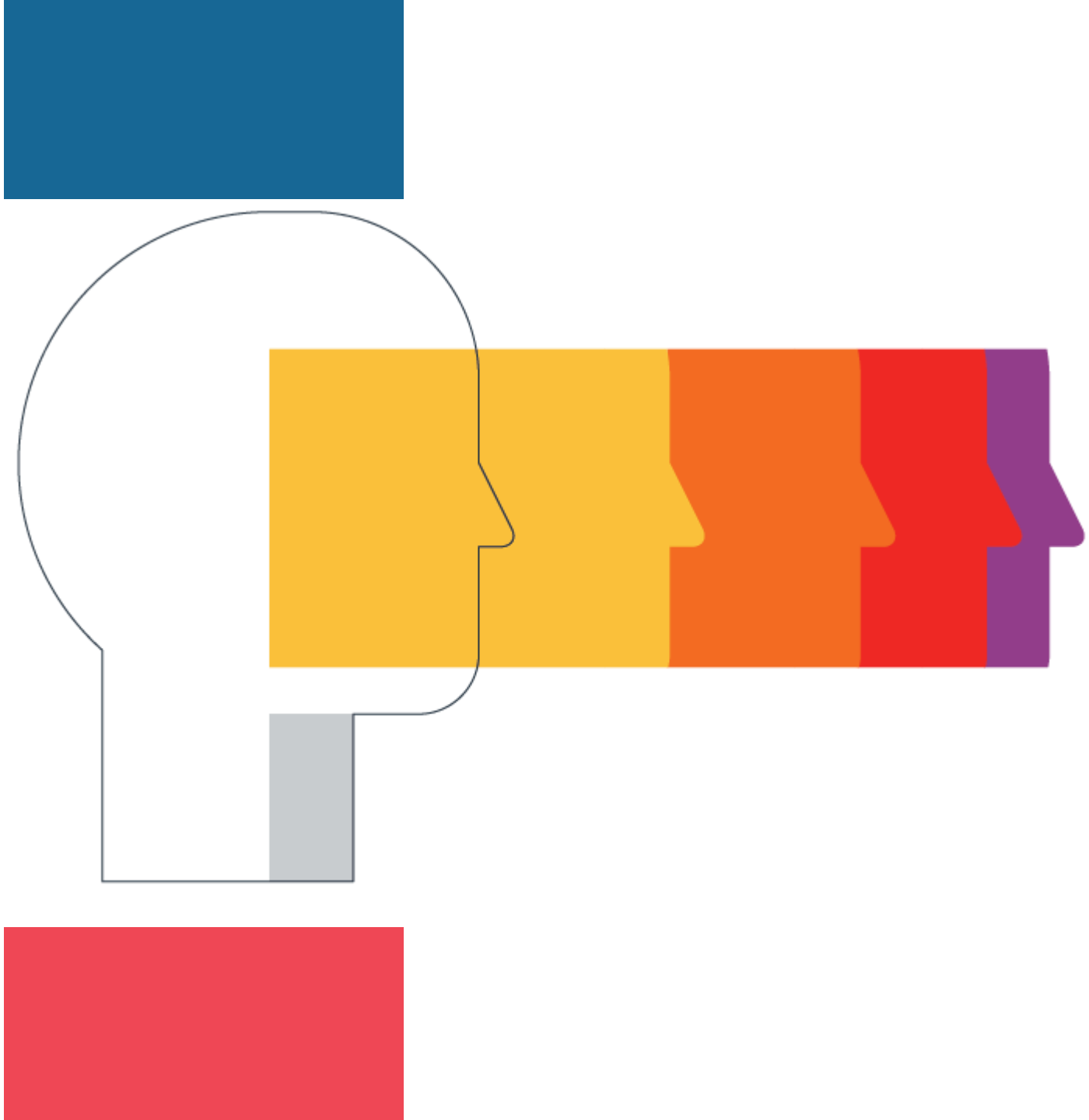
Speech Emotion Recognition

Prawesh Dahal



ELENE 6820

Final Project Presentation

May 15, 2019

Challenges in SER

- Which features are relevant and informative for SER?
- No explicit temporal boundaries of emotion states
- Emotion patterns vary across individuals

Objective

To compare the performance of conventional emotion classifiers (SVM) to RNN-based speech emotion recognition (SER)

Corpus

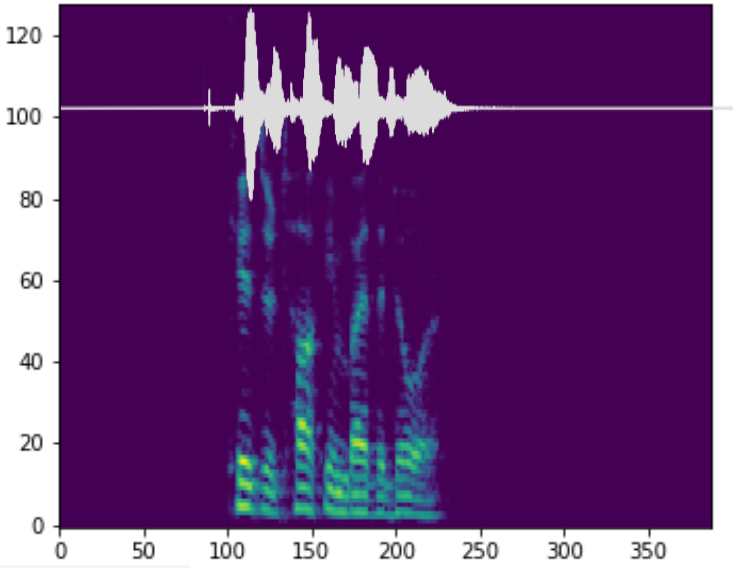
The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS)

1440 files, 12 female and 12 male actor speech recordings
8 emotions expressions - neutral, calm, happy, sad, angry, fearful, surprise, and disgust.

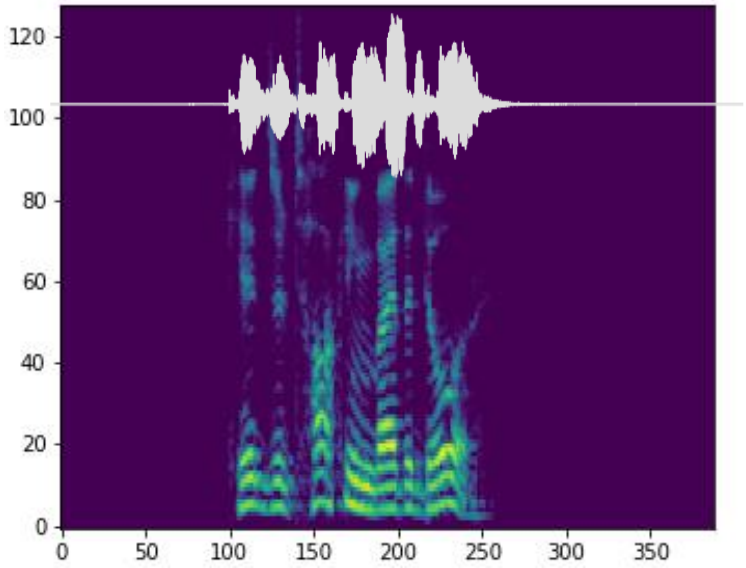
Reference Paper: Automatic speech emotion recognition using recurrent neural networks with local attention – Mirsamadi et al. (2017)

Sample traces and spectrogram

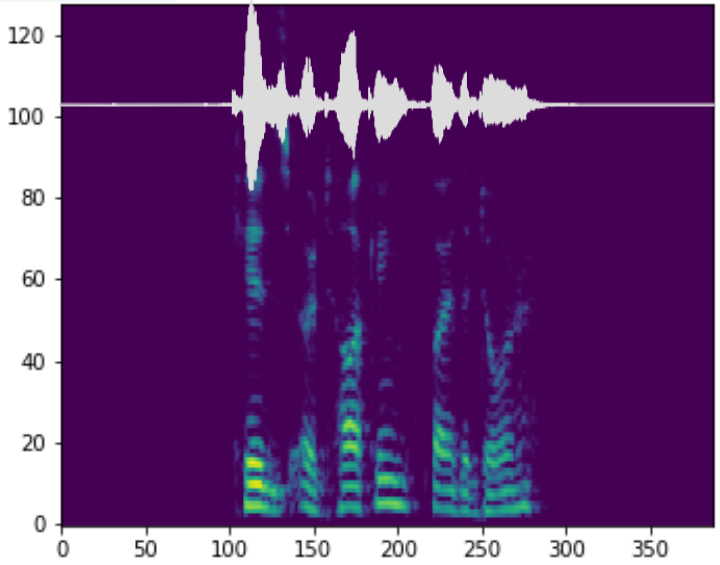
Neutral



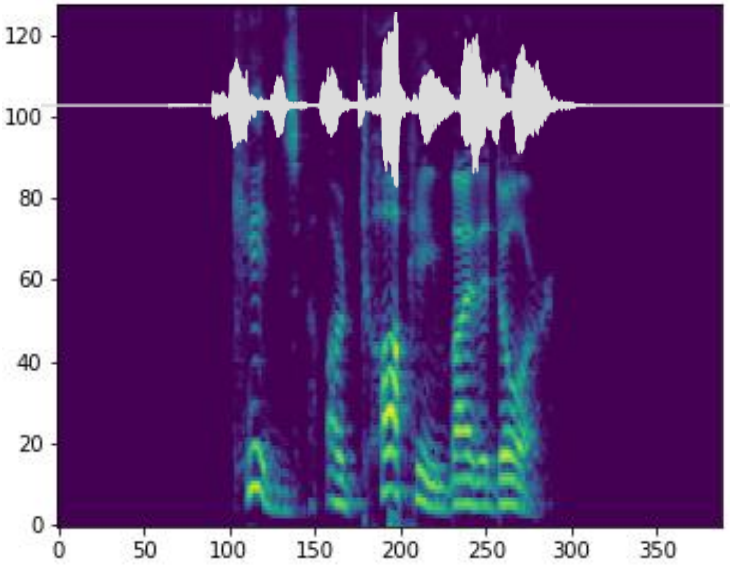
Happy



Sad



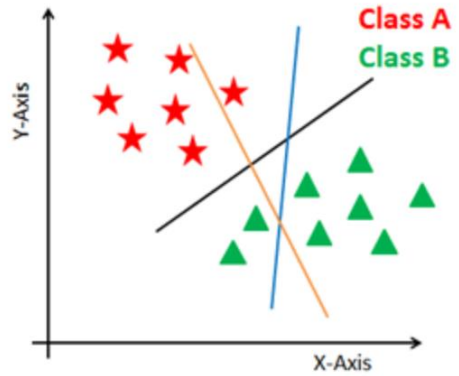
Angry

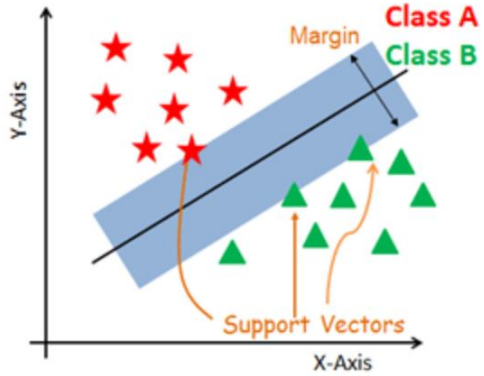


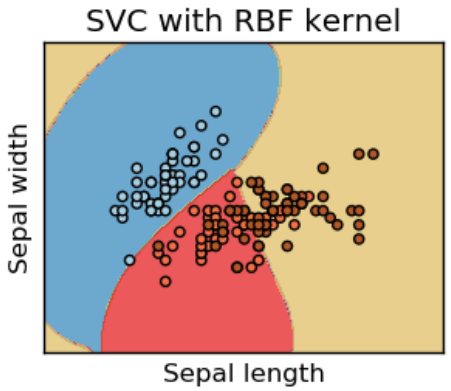
Statement - "Kids are talking by the door”

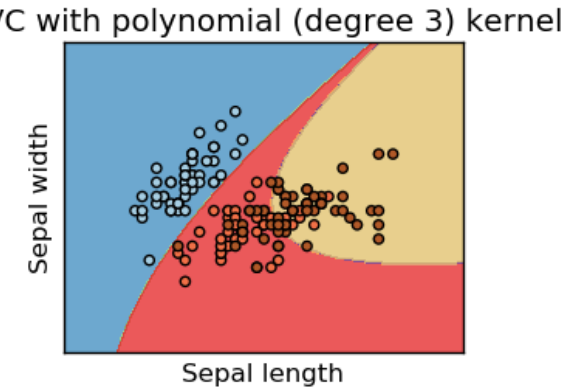
Classifier I: Support Vector Machine (SVM)

- Constructs a hyperplane in n-dimensional space to separate the classes
- Finds a hyperplane that maximizes the between class margin
- Kernel functions employed for nonlinear data









Classifier I: Support Vector Machine (SVM)

Feature Extraction I (LLD) → Feature Extraction II (HSF) → Normalization by global mean and standard deviation of neutral features → Train SVM Classifier Model → Emotions

Low-Level Descriptors (LLD)

- Short frames (50 ms)
- Hop length (10 ms)
- 17 LLDs
  - 13 MFCCs
  - Frame Energy
  - Spectral rolloff
  - Spectral centroid
  - Zero Crossing Rate

High-level statistical functions (HSF)

- Mean, std, min, max. etc

SVM Parameters

- Multiclass OneVsOne classifier
- Radial Basis Function (RBF) Kernel
- Tuned parameters using grid search method

Classifier I: Support Vector Machine (SVM) Results

Table 2. Accuracy comparison between hand-crafted and learned LLDs from raw spectral features.

Features	Classifier	HSFs	WA
emotion LLDs	SVM	Mean	53.3%
			61.2%

Literature Result

Project Result

Emotions	Neutral	Happy	Sad	Angry
Neutral	47.5	17.5	25	10
Happy	0	67.6	27	5.4
Sad	4.2	4.1	79.2	12.5
Angry	0	16.7	0	83.3

Classifier I: Support Vector Machine (SVM) Results

Table 2. Accuracy comparison between hand-crafted and learned LLDs from raw spectral features.

Features	Classifier	HSFs	WA
emotion LLDs	SVM	Mean	53.3%

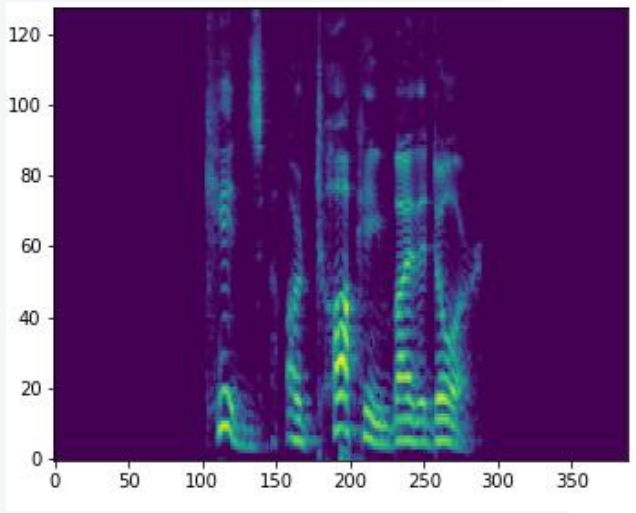
61.2% Literature Result

61.2% Project Result (4 emotions)

56.3% Project Result (8 emotions)

Emotion	Neutral	Calm	Happy	Sad	Angry	Fearful	Disgust	Surprise
Neutral	35.7	21.4	11.9	12.1	4.7	7.1	2.3	4.7
Calm	8.8	60	2.2	17.8	0	2.2	4.4	4.4
Happy	0	0	62.2	18.9	5.4	5.4	0	8.1
Sad	0	6.7	0	80	0	6.7	6.7	0
Angry	0	0	6.8	0	75.8	3.4	0	13.8
Fearful	0	2.9	8.8	11.8	5.9	52.9	5.9	11.7
Disgust	1.8	0	3.6	3.6	14.5	10.9	54.5	10.9
Surprise	0	2.7	8.3	2.7	8.3	19	8.3	50

Classifier II: LSTM



Train a LSTM model

Emotions

Mel spectrogram

- Train (70%), Validation (20%), Test (10%) data split from each emotion set
- One-hot target vectors for emotion classes
- Normalize

LSTM Parameters

- 128 memory cells
- Bi-directional
- Final frame (many-to-one) training
- Focal Loss

$$\mathrm{FL}(p_t) = -(1 - p_t)^{\gamma} \log(p_t)$$

- Local attention model (attention weights)

Classifier II: LSTM Results

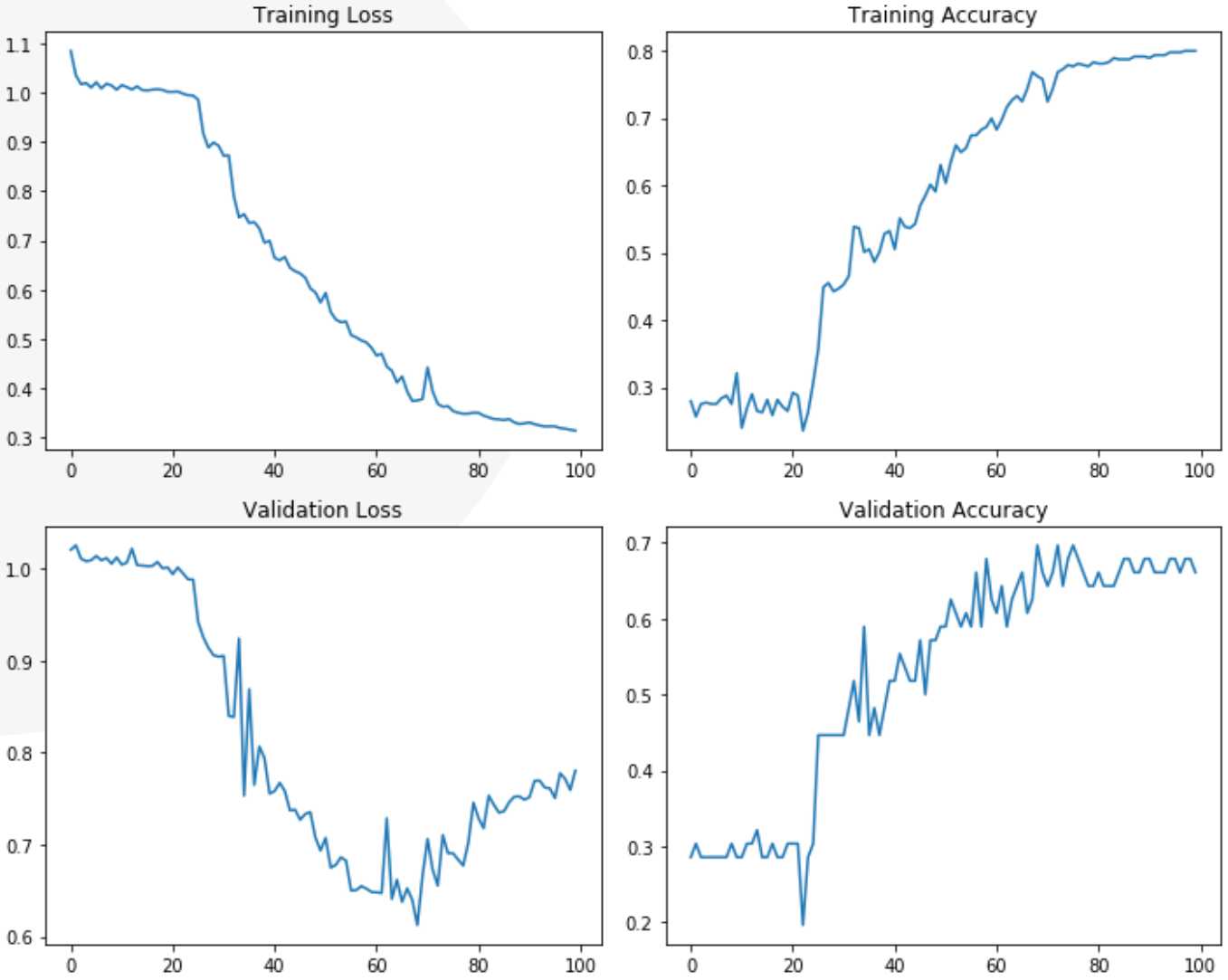


Table 3. Accuracy comparison between RNN architectures

Features	Temporal aggregation	WA
raw spectral	RNN–final frame	54.4%
		66.07%
		52.50%

```
model_LSTM = model_LSTM.eval()
test_LSTM = validate(model_LSTM, loader = test_loader)
# print('LSTM Test Accuracy: {:.3f}%'.format(test_LSTM))
```

```
    | end of validation epoch 100 | time:  0.23s | Accuracy 0.6607 |
-----------------------------------------------------------------------
```

```
model_LSTM = model_LSTM.eval()
test_LSTM = validate(model_LSTM, loader = test_loader)
# print('LSTM Test Accuracy: {:.3f}%'.format(test_LSTM))
```

```
    | end of validation epoch 100 | time:  0.37s | Accuracy 0.5250 |
-----------------------------------------------------------------------
```

Classifier II: LSTM WITH ATTENTION Results

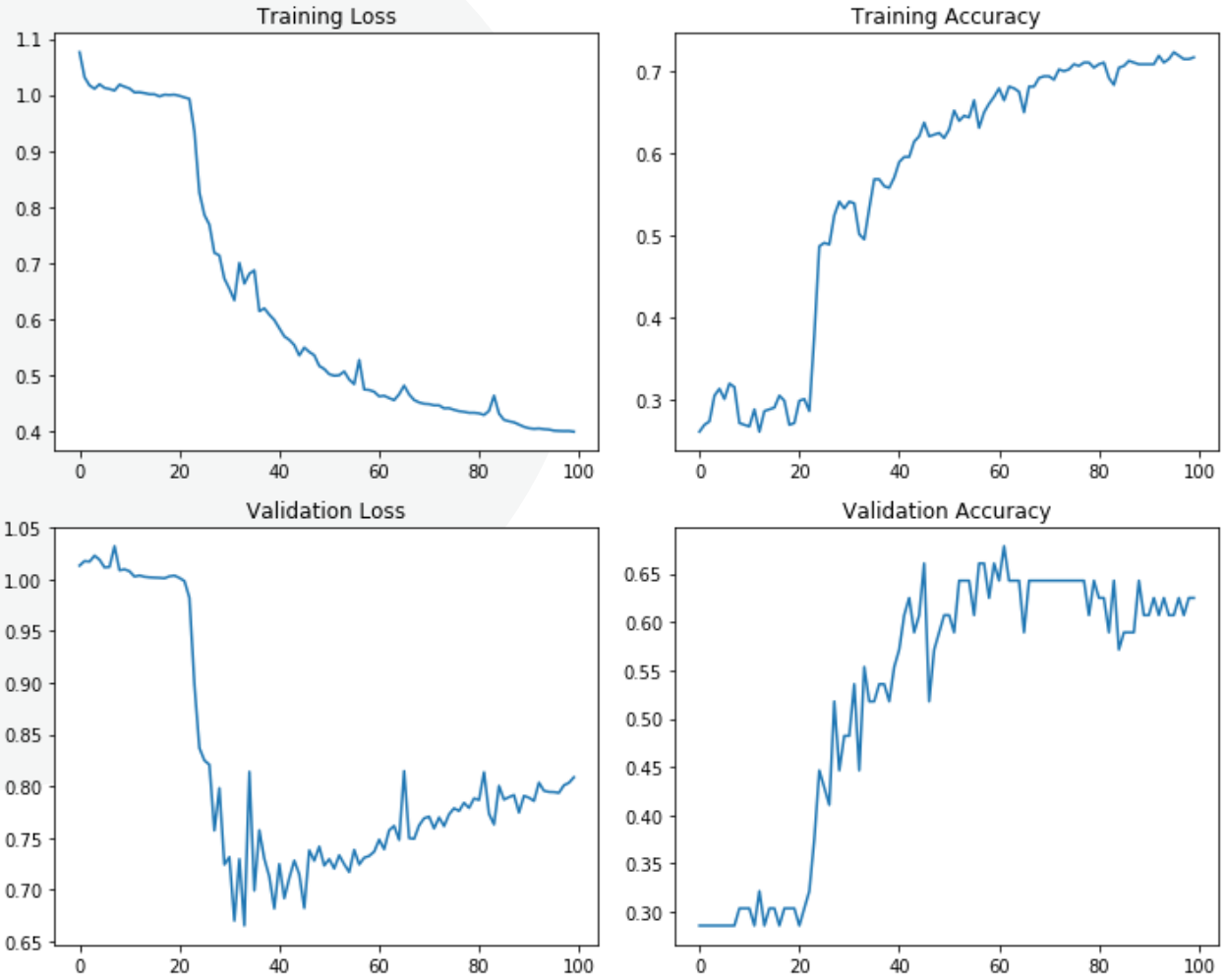


Table 3. Accuracy comparison between RNN architectures

Features	Temporal aggregation	WA
raw spectral	RNN–final frame	54.4%
	RNN with attention	61.8%

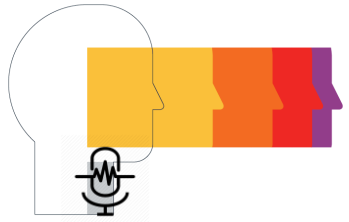
66.07%

67.86%

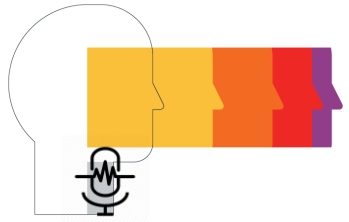
```
with open(args.model_save, 'rb') as f:
  model_LSTM_test = LSTM()
  model_LSTM_test.load_state_dict(torch.load(f))
model_LSTM_test = model_LSTM_test.cuda()
model_LSTM_test = model_LSTM_test.eval()
test_LSTM = validate(model_LSTM_test, loader = test_loader)
```

```
/usr/local/lib/python3.6/dist-packages/torch/nn/modules/rnn.py:54: Us
  "num_layers={}".format(dropout, num_layers))
    | end of validation epoch 100 | time:  0.21s | Accuracy 0.6786 |
-----------------------------------------------------------------------
```

Conclusion



- Conventional approaches in SER that uses machine learning algorithms such as SVM require hand-picked features.
- Deep learning architectures like RNN perform better compared to SVM.
- Attention mechanisms can be implemented within RNN that focuses on specific outputs of utterance that could be relevant determining emotions.



THANK YOU!