Improvements in Indirect Blood Pressure Estimation via Electrocardiography and Photoplethysmography

Andrew Stirn
astirn@stanford.edu

***Abstract*—With the rapid adoption of wearable technology, a ubiquitous platform may exist for continuous cuff-less blood pressure monitoring. Devices that monitor photoplethysmography (PPG) and/or electrocardiography (ECG) signals may be capable of accurately estimating blood pressure. Previous work has demonstrated promising estimation techniques, but they remain infeasible for consumer adoption due to their need for per-subject calibration and/or lack of performance. This paper explores improvements to these approaches by expanding the feature space and exploring more expressive models that estimate mean arterial, systolic, and diastolic blood pressures. These models are trained with features and targets derived from 12,000 sets of ECG, PPG, and blood pressure time-series data. Results indicate that the additional features had a positive impact on results but less so than increasing model complexity. While significant improvements were accomplished, all models struggled to estimate systolic blood pressure to a commercially viable standard. Results indicate that future work should assess convolutional and/or recurrent neural networks as candidates for this estimation problem.**

I. Introduction

Recent estimates attribute nine million deaths annually to elevated blood pressure [1]. The World Health Organization claims early detection of hypertension is critical to minimizing risk of heart attack, heart failure, stroke, and kidney failure [1]. In 2014, 20% of Americans owned a wearable device. Wearable adoption rates are tracking to those of tablets, of which 40% of Americans now own [2]. Wearable devices have a myriad of sensors, two of which have potential to provide cuff-less and non-invasive blood pressure monitoring: electrocardiography (ECG) and photoplethysmography (PPG) sensors. ECG is voltage measure of heart activity collected at the chest. PPG is a blood flow measurement taken optically at the finger. Blood pressure is a pressure measurement conducted externally with an inflatable cuff or invasively with a catheter—these direct methods are not practical for widespread continuous monitoring. The pervasive rise of wearable technology and the health benefits bestowed by blood pressure awareness and monitoring compels advancing current estimation techniques.

Modeling blood pressure requires an algorithm that ingests ECG and/or PPG data. This data can be the complete time-series and/or derived features. The algorithm(s) then must output estimates for systolic, diastolic, and mean arterial blood pressure in mmHg. This paper exclusively employs derived features as inputs, but concludes that employing models that ingest time-series directly such as convolutional neural networks may offer further performance gains. Because the outputs are continuous, this paper explores the performance of two linear output model families: linear regression and linear output neural networks. For the neural networks, this paper assesses performance for varying network depths and neuron populations.

II. Related Work

Initial blood pressure estimation attempts relied exclusively on pulse-transit time, the time from ECG R peak to PPG wavefront arriving at the fingertip [3]. This methodology exploits the relationship between blood wave velocity and pressure. However, this approach requires frequent per-subject calibration making it unsuitable for commercial deployment. Specifically, computing fluid velocity from time requires distance and other parameterizations. Newer work utilizes recurrent neural networks that ingest only PPG data to estimate blood pressure waveforms [4]. While their results are impressive, their approach trained a network for each subject using invasively collected data. As such this approach has no generalization guarantees and cannot be easily deployed rendering it commercially unviable.

Other work utilizes pulse-transit time and additional features derived from ECG and PPG signals in lieu of per-subject calibration [5]. This work demonstrates promising results for models that generalize to multiple subjects. However, this paper posits there is room for improvement. First, their feature space is limited to just ten features. Second, they only tested models with limited ability to express highly non-linear systems: linear regression, support vector regression with an RBF kernel, and a single hidden layer neural network with a linear output. The methods and results presented in this paper investigate improvements in these two areas.

III. Data Set and Features

The data set [6] comprises 12,000 sets of variable length ECG, PPG, and blood pressure waveform data sampled at 125Hz. The ECG and PPG signals are the basis for the feature space. The blood pressure waveform provides computable targets for systolic, diastolic, and mean arterial blood pressure. The models employed in this paper utilize features and targets generated on a per-beat basis. Accordingly, feature and target generation comprises several steps:

1. Classifying R peaks on the ECG signal.
2. Filtering misclassified R peaks.
3. Locating resulting PPG and blood pressure wavefronts for all remaining R peaks.
4. Feature and target computation for all wave segments.

The first step is a combination of signal processing and unsupervised learning. ECG waveforms can vary substantially across subjects and time. In some cases, the R peak does not necessarily correspond to the largest local peak. Additionally, bad electrode connections can produce high frequency noise. Traditional high-pass filtering with naive peak detection can break down during these challenging scenarios. The signal processing step, inspired by [7], involves applying a wavelet transform, equation (1) with S = 8, to the ECG signal. In this implementation, the wavelet was convolved with the signal in both forward and reverse directions to guarantee zero-phase filtering for all potential wavelets. The resulting values for each wavelet scale, s, overlaid with the original ECG signal can be seen in the first subplot of figures 1 and 2. Thereafter, all positive and negative ECG peaks are identified using derivative tests. A positive peak is defined to have a positive derivative to the left and a derivative greater than or equal to zero to the right. Polarity is reversed for finding negative peaks. Each positive peak was assigned 16 (2xS) features: its own wavelet transform and the closest-to-the-right negative peak's wavelet transform values. Each subject had those features presented to the k-means algorithm for k = 2. However, instead of initializing the two centroids randomly, they were initialized with the the samples whose feature vectors had the largest and smallest l-2 norms. Random initialization occasionally could prevent k-means from successfully isolating R peaks. This deterministic initialization encouraged maximal separation of a desired feature and eliminated many misclassifications encountered with random initialization. One drawback of deterministic initialization occurs if an ECG waveform contains an anomalously large noise impulse. In this scenario, k-means can incorrectly identify the impulse as the only R peak. However, misclassification of R peaks due to random centroid initialization was more prevalent and damaging to the final objective of the algorithm. Once k-means converged, any peak belonging to the centroid with the largest norm was labeled an R peak. The resulting labels are shown on top of the original ECG signal in the second subplots of figures 1 and 2.

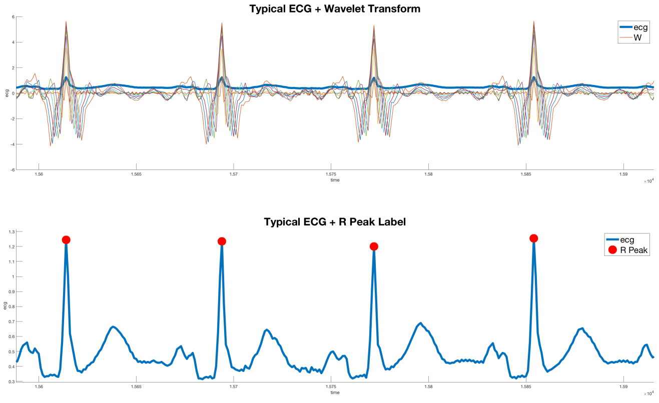

Figure 1: Typical ECG Waveform

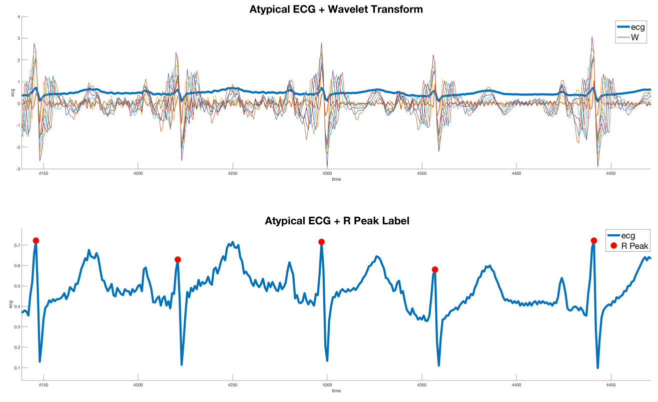

Figure 2: Atypical ECG Waveform

Wavelet Operator:

$$w[s] = 2\delta[0] - \delta[-s] - \delta[+s]$$

$$W[s,t] = \sum_{k=-s}^{s} \text{ecg}[t-k]w[k]$$

$$= 2\ \text{ecg}[t] -\ \text{ecg}[t-s] -\ \text{ecg}[t+s]$$

$$W \ \epsilon\ \mathbb{R}^{S\times T}$$

Equation 1: Wavelet Transform

$$BPM = \frac{60 \times 125}{RR_{\text{interval}}}$$

Equation 2: Computing BPM

Weighted Least Squares with RBF Kernel:

$$\beta = (X^T W X)^{-1} X^T W y$$

$$\hat{y} = X\beta$$

$$X = \begin{bmatrix} t_1 & 1 \\ \vdots & \vdots \\ t_T & 1 \end{bmatrix} \ \epsilon\ \mathbb{R}^{T\times 2},\ \beta\ \epsilon\ \mathbb{R}^{2\times 1}$$

$$w_i = \exp(\frac{-(t_{\text{querry}} - t_i)^2}{2\tau^2})$$

$$W = \text{diag}(w_1, \ldots, w_T)\ \epsilon\ \mathbb{R}^{T\times T}$$

Equation 3: Weighted Least Squares

A heart beat generates a PPG and blood pressure wavefront. Both PPG and blood pressure wavefronts begin sometime after the initial R peak and finish sometime between the next two consecutive R peaks. Thus, feature extraction requires windowing PPG data across two RR intervals. To avoid erroneous feature calculations from incorrect windowing, RR intervals receive additional filtering. This filtering step involves computing a user's heart-rate in beats-per-minute (BPM) from the RR interval in accordance with equation (2). These values are then detrended versus a weighted least squares (WLS) estimate, equation (3). The weights are computed via a RBF kernel with a bandwidth parameter of 150. The detrended signal is then the difference between each BPM measurement and its WLS estimate at that point. The resulting signal allows for easy outlier detection. Any detrended BPM whose absolute value is greater than 5 is removed. Figure 4 illustrates this process in full. The first subplot shows BPM values and their WLS estimates. The second subplot has the detrended signal with +/-5 BPM lines shown. The third subplot displays only the BPM values that pass the filter.

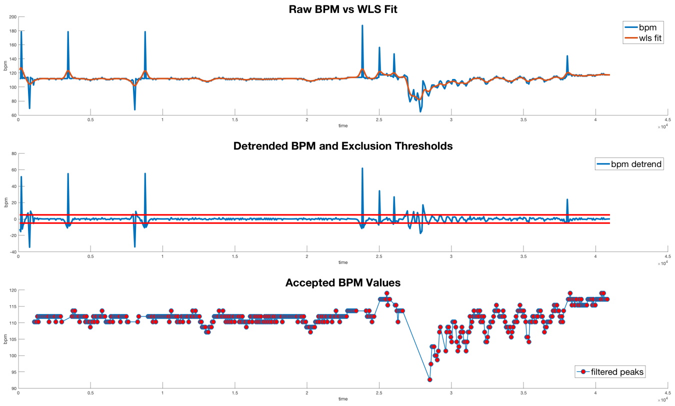

Figure 4: WLS R-Peak Filter

Identification of the PPG and blood pressure (BP) wavefronts requires examining the signals across two RR intervals as depicted by those times between the three green dots on the ECG subplot of figure 5. The method for extracting corresponding PPG and blood pressure wave segments is identical:

1. Find start time: lowest negative peak on the first RR interval (first red dot in figure 5 subplots for PPG and BP).
2. Find stop time: lowest negative peak on the second RR interval (last red dot in figure 5 subplots for PPG and BP).
3. Find the peak: maximum value between start and stop times (green dot in figure 5 subplots for PPG and BP).

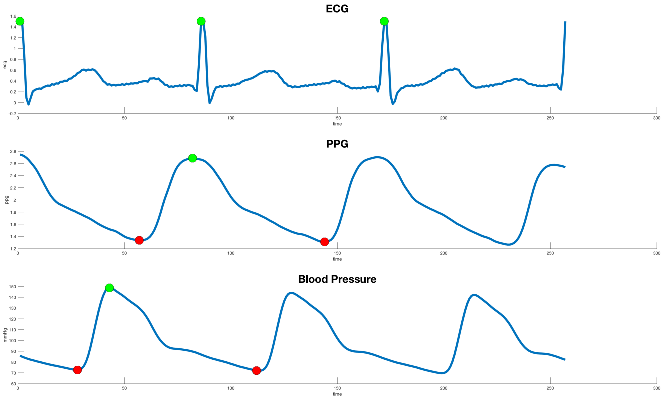

Figure 5: PPG and Blood Pressure Wavefront Identification

With feature windows assembled, feature and target computation can begin. The proposed feature space includes twenty (N = 20) derived features comprised of three groups. All features except the first can be seen in figure 6.

I. Time based features to represent pulse transit time and subject parameters:

1. Subject's heart rate.
2. Time from R peak to PPG wavefront's start.
3. Time from R peak to PPG wavefront's end.
4. Time from R peak to PPG wavefront's max slope.
5. Time from R peak to PPG wavefront's peak.
6. Time from PPG peak to PPG inflection point.

II. Normalized PPG wavefront features to represent subject parameters:

7. Max slope of PPG wavefront.
8. PPG integration: from start to maximum slope.
9. PPG integration: from maximum slope to peak.
10. PPG integration: from peak to inflection point.
11. PPG integration: from inflection to the end.

III. PPG integral quantiles to represent subject parameters:

12-20. The time for PPG integration to reach 10, 20, … , and 90 percent of total integration.

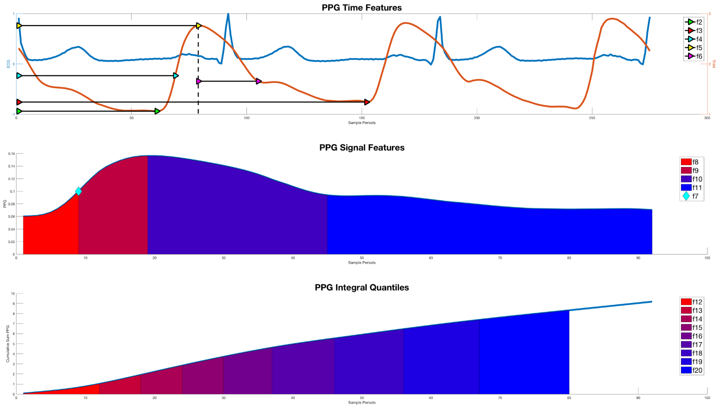

Figure 6: Proposed Features

This proposed feature space includes those from [5] save two: the height of the dicrotic notch and the ratio of peak and dicrotic notch heights. These features were omitted as the dicrotic notch is not detectable for all subjects and the inflection point is a similar substitute. Features 3, 7, and 12-20 are those additional to [5]. For comparison, the ten features of [5] were also computed, but use the inflection point instead of the dicrotic notch. It is left to the reader to reference their work. The three blood pressure targets are defined as:

1. Systolic pressure is a wavefront's max pressure.
2. Diastolic pressure is wavefront's final pressure.
3. Mean arterial pressure is wavefront's mean pressure.

IV. Methods

Given the targets are continuous, two linear output model families are considered: linear regression and linear output neural networks (NN). The neural networks employ hyperbolic tangent (tanh) activations as neurons. By this paper's convention, the number of neurons per hidden layer, M, is ratiometric to the number of input features, N. The five models trained and tested on the expanded feature space (N = 20) and feature space of [5] (N = 10) are:

1. Linear regression: equation (4), figure 7.
2. Single hidden layer NN with M = N tanh activations per layer: equation (5), figure 8.
3. Single hidden layer NN with M = 2N tanh activations per layer: equation (5), figure 8.
4. Dual hidden layer NN with M = N tanh activations per layer: equation (6), figure 9.
5. Dual hidden layer NN with M = 2N tanh activations per layer: equation (6), figure 9.

After the preprocessing and feature extraction steps, 2.6 million samples resulted from the 12,000 sets of data. However, due to computational limitations only those samples corresponding to the first 3000 sets were used. This redaction

Linear Regression:

$$y = w^T x + b$$

$$w, x\ \epsilon\ \mathbb{R}^N$$

$$b\ \epsilon\ \mathbb{R}$$

Equation 4: Linear Regression

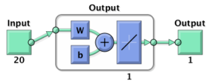

Figure 7: Linear Regression

Single Layer Linear Output NN with M tanh Neurons per Layer:

$$y = w^T \tanh(W_1 x + b_1) + b$$

$$x\ \epsilon\ \mathbb{R}^N,\ W_1\ \epsilon\ \mathbb{R}^{M \times N}$$

$$w, b_1\ \epsilon\ \mathbb{R}^M,\ b\ \epsilon\ \mathbb{R}$$

Equation 5: Single Hidden Layer NN

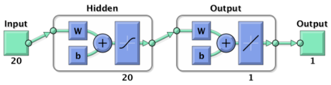

Figure 8: Single Hidden Layer NN w/ M = 20

Dual Layer Linear Output NN with M tanh Neurons per Layer:

$$y = w^T \tanh(W_2 \tanh(W_1 x + b_1) + b_2) + b$$

$$x\ \epsilon\ \mathbb{R}^N,\ W_1\ \epsilon\ \mathbb{R}^{M \times N},\ W_2\ \epsilon\ \mathbb{R}^{M \times M}$$

$$w, b_1, b_2\ \epsilon\ \mathbb{R}^M,\ b\ \epsilon\ \mathbb{R}$$

Equation 6: Dual Hidden Layer NN

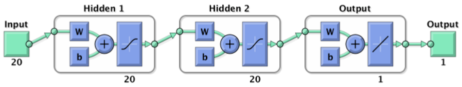

Figure 9: Dual Hidden Layer NN w/ M = 20

amounted to 604,052 training samples. These features had min/max mapping applied to support regularization. This pre-training step is a linear mapping of each feature to the interval: [-1,1]. This mapping ensures even weight penalization during regularization despite varying magnitudes of the original features. All models were trained using Matlab's Neural Network Toolbox and the Levenberg–Marquardt algorithm [8] to minimize mean square error with 70% of data for training, 15% for cross-validation, and the 15% for testing.

Linear regression learns a set of input weights and a bias that minimizes mean square error between its output and the training targets. It is the least expressive model tested as the output can only assume a linear combination of the inputs. As such, this model struggles to model non-linear relationships. Conversely, neural networks can learn non-linear relationships. Given the non-linear relationship between the feature and target spaces [5], this ability is desired. This non-linear expressive power results from the activation functions (neurons) in the hidden layer. The networks used in this paper all employ the hyperbolic tangent function: a non-linear mapping of a scalar input to the interval [-1,1]. Each neuron's input is a linear combination of the previous layer's outputs. Increasing the number of neurons in or the depth of a neural network allows for more complex non-linear relationships to be modeled. For the neural networks above, the output layer is just a linear combination of the final hidden layer's outputs. It should be noted that neural networks are prone to overfitting compared to less complex models like linear regression. Intuitively this makes sense as linear regression only learns a weight vector and bias scalar. The neural networks listed above learn an output weight vector, an output bias scalar, and an additional weight matrix and bias vector per hidden layer. Accordingly, close attention must be paid to differences between training and test errors to avoid overfitting.

V. Results

The models were assessed using root mean square error (RMSE) and three metrics from the British Hypertension Society: percent of readings within 5, 10, and 15 mmHg of truth. Tables 1-3 present training and test results for mean arterial (MA), systolic, and diastolic blood pressures using the new feature space (N = 20). Tables 4-6 present the same results for the old feature space (N = 10). For both feature spaces, training and test errors were nearly equivalent suggesting that overfitting is not a problem. For visual comparison, figure 10 plots the model types in increasing complexity versus the test set performance measures for the new feature space and the features space of [5].

The neural networks tested in [5] had a single hidden layer with 5-15 neurons. It is unclear how many neurons were in the network for which they reported results. Since it makes sense to list the top performer and that the results of this paper demonstrate performance increases with neuron population, this paper will assume their results are for 15 neurons. The authors of [5] reported RMSE's for this network as 8.84, 13.78, and 6.86 respectively for mean arterial, systolic, and diastolic pressure estimation. These numbers closely match the test set RMSE performance for the NN 1x10 and NN 1x20 models in tables 4-6.

Table 1: MA BP Estimation Results for Expanded Feature Space (N = 20)

	Percent within 5 mmHg		Percent within 10 mmHg		Percent within 15 mmHg		RMSE	
Data Set	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**
LinReg	31.21	31.10	58.90	58.63	78.65	78.56	9.80	9.83
NN 1x20	43.21	43.04	71.83	71.68	86.72	86.59	7.71	7.73
NN 1x40	46.28	45.66	74.38	74.04	88.49	88.36	7.24	7.30
NN 2x20	47.86	47.89	75.92	75.98	89.45	89.41	6.99	7.01
NN 2x40	54.32	54.07	80.82	80.49	92.02	91.73	6.13	6.21

Table 2: Systolic BP Estimation Results for Expanded Feature Space (N = 20)

	Percent within 5 mmHg		Percent within 10 mmHg		Percent within 15 mmHg		RMSE	
Data Set	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**
LinReg	19.06	19.08	37.27	37.29	53.88	53.98	16.14	16.15
NN 1x20	28.47	28.53	51.73	51.75	68.59	68.66	12.51	12.50
NN 1x40	30.73	30.77	54.54	54.72	71.00	71.05	11.77	11.79
NN 2x20	33.25	33.12	57.75	57.38	73.83	73.36	11.09	11.17
NN 2x40	38.28	37.79	63.52	62.78	78.50	77.80	9.88	10.09

Table 3: Diastolic BP Estimation Results for Expanded Feature Space (N = 20)

	Percent within 5 mmHg		Percent within 10 mmHg		Percent within 15 mmHg		RMSE	
Data Set	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**
LinReg	38.52	38.53	70.18	70.00	89.15	89.19	7.99	8.00
NN 1x20	49.20	49.11	78.41	78.39	91.53	91.60	6.68	6.69
NN 1x40	53.08	52.95	81.20	81.13	92.74	92.68	6.22	6.24
NN 2x20	55.96	56.17	82.88	82.88	93.30	93.35	5.90	5.88
NN 2x40	61.11	60.80	85.87	85.50	94.56	94.31	5.31	5.40

Table 4: MA BP Estimation Results for Original [5] Feature Space (N = 10)

	Percent within 5 mmHg		Percent within 10 mmHg		Percent within 15 mmHg		RMSE	
Data Set	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**
LinReg	28.25	28.32	55.47	55.41	77.55	77.42	10.23	10.24
NN 1x10	37.28	37.39	65.80	65.99	83.33	83.48	8.69	8.66
NN 1x20	41.45	41.75	70.04	70.12	85.76	85.70	8.01	7.99
NN 2x10	42.87	42.83	71.18	70.89	86.42	86.31	7.80	7.84
NN 2x20	46.69	46.81	74.68	74.83	88.82	88.90	7.18	7.16

Table 5: Systolic BP Estimation Results for Original [5] Feature Space (N = 10)

	Percent within 5 mmHg		Percent within 10 mmHg		Percent within 15 mmHg		RMSE	
Data Set	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**
LinReg	17.61	17.67	35.62	35.42	52.05	51.91	16.52	16.56
NN 1x10	25.35	25.17	47.29	47.31	64.34	64.24	13.57	13.59
NN 1x20	27.39	27.30	50.36	50.40	67.17	67.26	12.83	12.86
NN 2x10	27.95	27.78	50.78	50.57	67.19	67.14	12.81	12.84
NN 2x20	32.19	32.54	55.94	56.29	72.20	72.50	11.47	11.41

Table 6: Diastolic BP Estimation Results for Original [5] Feature Space (N = 10)

	Percent within 5 mmHg		Percent within 10 mmHg		Percent within 15 mmHg		RMSE	
Data Set	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**
LinReg	35.19	35.26	67.89	67.91	89.01	89.05	8.30	8.29
NN 1x10	45.38	45.50	75.24	75.35	90.40	90.45	7.18	7.16
NN 1x20	47.81	47.67	77.83	77.72	91.45	91.40	6.82	6.84
NN 2x10	49.22	49.21	78.34	78.28	91.46	91.40	6.69	6.71
NN 2x20	54.40	54.56	81.79	82.00	92.89	92.88	6.09	6.08

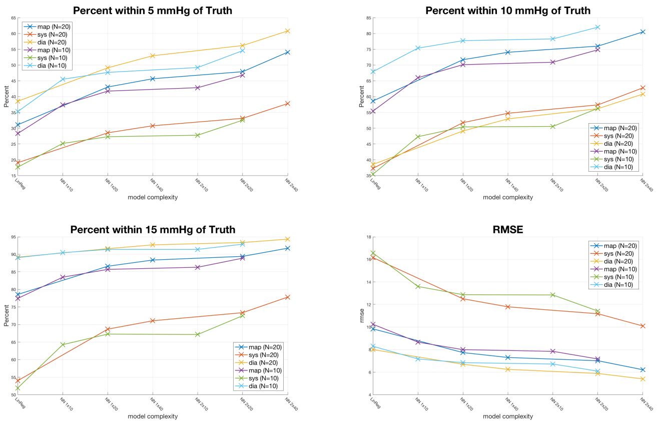

Figure 10: Comparative Results

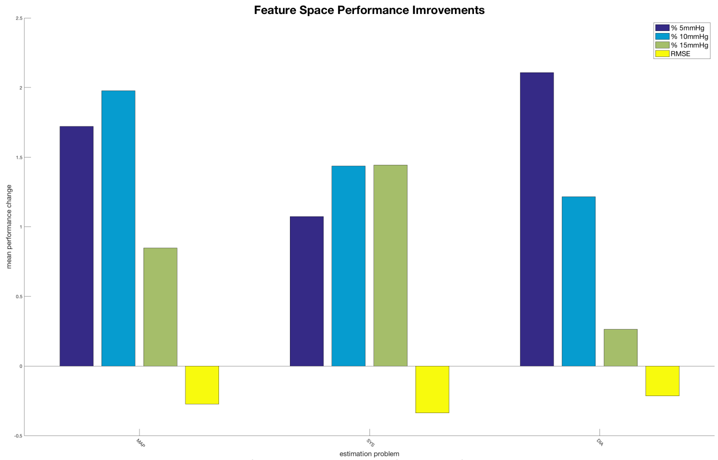

Figure 11: Comparative Results

Table 7: Comparison with BHS Standard and Previous Work [5]

	Percent within 5 mmHg	Percent within 10 mmHg	Percent within 15 mmHg	RMSE mmHg
BHS Grade A Device Standard	60.00	85.00	95.00	N/A
BHS Grade B Device Standard	50.00	75.00	90.00	N/A
BHS Grade C Device Standard	40.00	65.00	85.00	N/A
MA Pressure Results of [5]	44.70	71.60	86.70	7.52
MA Pressure ANN 2x40	54.07	80.49	91.73	6.21
Sys. Pressure Results of [5]	28.80	51.50	69.50	12.38
Sys. Pressure ANN 2x40	37.79	62.78	77.80	11.17
Dia. Pressure Results of [5]	51.20	78.90	93.60	6.34
Dia. Pressure ANN 2x40	60.80	85.50	94.31	5.40

Comparing the performance between the two feature spaces, it seems the new feature space was responsible marginal improvements in performance. Note that the two feature spaces had three overlapping models: linear regression, NN 1x10, and NN 1x20. The mean performance delta for these three models between the two feature spaces is a measure to assess the impact of the expanded feature space. Figure 11 plots this performance delta for the four performance metrics and the three estimation problems. For mean arterial and diastolic pressure estimation, the new feature space had the most substantial improvements to low scale error (percent within 5 and 10 mmHG of truth). Conversely, the new features had a more balanced improvement to systolic pressure estimation.

The new feature space coupled with the most expressive model, a dual hidden layer neural network with 40 neurons per hidden layer, was the top performing estimator for all three estimation problems. Its test set results as compared to the BHS standards and the results of [5] are in table 7. The performance for the expanded feature space and this model attained a grade B for mean arterial pressure and grade A/B for diastolic pressure estimation. Unfortunately while systolic estimation saw large improvements, it remained just shy of grade C status.

VI. Conclusion

In summary, the combination of the expanded feature space and higher model complexities offered substantial improvements to mean arterial, systolic, and diastolic blood pressure estimation as compared to [5]. Running their feature space on similar sized neural networks yielded very similar RMSE performance for all three estimation problems. This similarity suggests that the differing preprocessing steps and data redaction of this paper had little impact on performance. In isolating the two proposed improvements to [5], expanded feature and model spaces, it appears that the expanded model space was responsible for the majority of the performance gains. Comparing the performance of the three estimation problems, models have a much harder time estimating systolic than diastolic pressures. Because mean arterial pressure incorporates systolic and diastolic readings, it makes intuitive sense that its performance lies in the middle.

To obtain commercial viability, it is posited that systolic performance must attain at least grade C. Because test and training errors are so similar, it seems the models suffer from high bias, which could result from too few and/or irrelevant features. Additionally, the employed models may be unable to accurately represent the highly non-linear relationships between features and targets. Examining model complexity versus performance in figure 10, it does not appear an asymptote has been hit. As such it seems safely appropriate to move to more complex models, such as convolutional neural networks with and without recurrent activations. These model families obviate the need for feature generation and allow the models to determine the most relevant feature space. Additionally, these models are highly compatible with time series data, which eliminates the need for the complex windowing schemes in section III. However, to train and assess these models across the full data set requires vastly more computational resources. As Matlab on a desktop PC struggled to train the comparatively smaller networks on 2.6e6 training samples—leading to the data redaction mentioned in section IV—future work should employ a toolset that can easily deploy to the cloud for parallelized CPU and GPU resources. Finally, the health benefits and market potential for a blood pressure wearable coupled with the promising results of this paper and those referenced compels further investigation and development.

VII. References

[1] World Health Organization. "A global brief on Hypertension: Silent killer, global health crisis," in *World Health Day*, 2013.

[2] PWC. "The Wearable Future," in *Consumer Intelligence Series*, 2014.

[3] M. Wong, C. Poon, Y. Zhang, "An evaluation of the cuffless blood pressure estimation based on pulse transit time technique: a half year study on normotensive subjects," in *Cardiovascular Engineering*, vol. 9, no. 1, pp. 32–38, 2009.

[4] C. Sideris, H. Kalantarian, E. Nemati, M. Sarrafzadeh, "Building Continuous Arterial Blood Pressure Prediction Models Using Recurrent Networks," in *Smart Computing, An IEEE conference on*, 2016.

[5] M. Kachuee, M. M. Kiani, H. Mohammadzade, M. Shabany, "Cuff-Less High-Accuracy Calibration-Free Blood Pressure Estimation Using Pulse Transit Time," in *IEEE International Symposium on Circuits and Systems (ISCAS'15)*, 2015.

[6] M. Kachuee, M.M. Kiani, H. Mohammadzade, M. Shabany. (2015). *Cuff-Less Blood Pressure Estimation Data Set* [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Cuff-Less+Blood+Pressure+Estimation

[7] F. Scholkmann, J. Boss, M. Wolf. "An Efficient Algorithm for Automatic Peak Detection in Noisy Periodic and Quasi-Periodic Signals," in *Algorithms—Open Access Journal*, 2012.

[8] MATLAB and Neural Network Toolbox Release 2016b, The MathWorks, Inc., Natick, Massachusetts, United States.