A Novel Method for Gait Segmentation Using Low Dimension Output Embedding Mapping in Convolutional Neural Network

Xueyan Li

Supervisor: Dr Lance Rane

***Abstract*—Wearable devices and active prostheses that deliver therapy or modulate gait characteristics in coordination with the gait cycle must be capable of identification and classification of gait events in real time. Artificial neural networks that achieve good accuracy have been successfully employed for gait classification. However, there is a lack of studies on the generalizability of gait classification algorithms to different subjects, especially those with pathological gait. At the same time, due to high computational cost and complexity, many of these models are not suitable for use on low powered devices for real time gait analysis. This paper describes the development of a lightweight convolutional neural network that can be deployed to portable hardware for accurate gait phase classification in patients with osteoarthritis. To achieve high performance and efficiency, a novel approach based on output embedding mapping is used to encode an inductive prior about the physical nature of the gait cycle, enabling improved generalisation performance on pathological gait at the cost of a few additional parameters. A lightweight 15KB model achieves 84.9% and 67.0% accuracy classifying 8 gait phases for healthy and patients with osteoarthritis with 2.76ms inference latency.**

1. Introduction

Functional Electrical Stimulation (FES) can treat neurological disorders such as drop foot [1] following stroke by applying low-energy electrical pulses to certain areas of the body to generate muscle contraction. Small wearable FES devices allow patients to receive continuous treatment outside of clinical settings and are essential for patients who rely on FES to perform day-to-day activities [2]. Effective usage of FES requires precise delivery of stimulation that coincides with the intended movement in real time [3] so that side effect resulting from over stimulation is minimized, and actions of individual muscle groups are more precisely targeted.

To precisely control FES, one must obtain information on the subject's movement status in real time. Such information may be provided by pressure sensors mounted in the shoes of the subject, providing a signal indicating the current status of gait [4]. Gait can be divided into swing and stance phase [5]. Stance phase begins when the heel strikes the ground and ends when the toe leaves the ground. However, such foot sensors are inconvenient for patients and typically provide only limited binary stance and swing phase information, which is insufficient for precise and complex control algorithms [6].

Several alternative sensor systems have been explored to obtain more detailed motion information, with inertial measurement-based systems emerging as leading candidates. These sensors convey information on linear acceleration and angular velocity of body segments, via onboard Microelectromechanical Systems (MEMS) accelerometers and gyroscopes [7]. Inertial Measurement Unit (IMU) signals usually involve angular, linear acceleration, and sometimes magnetic field intensity information in multiple axis. Coupled with high frequency time series format, IMU signals result in high-dimensional noisy data, that makes machine learning methods good candidates to process such data. In particular, Convolutional Neural Networks (CNN) have been an area of interest for gait segmentation due to their ability to detect translationally-invariant features [8], [9]. However, such accurate models come with the cost of large number of parameters, which makes them limited to offline use [10], [11].

For clinical use, the model must be deployed in real time on portable, battery-powered hardware, placing constraints on computational power and memory footprint. Such constraints may be relieved by the streaming and remote processing of data but this incurs a latency cost that is often unacceptable, and may evoke privacy concerns, particularly where sensitive medical data are streamed. We are therefore concerned with a fully embedded solution where all data processing and inference takes place at the edge. A typical 32-bit microcontroller such as the STM32H743ZI2 has 2 MB of flash memory and operates at a frequency up to 400 MHz. A common 8-bit microcontroller may have 4 to 32 KB of memory with 8 to 32 MHz frequency. These devices have very limited memory and much lower clock speed than a typical GPU. Thus, to achieve good accuracy and low latency on an embedded device, the model must be optimized in terms of size and inference speed. Battery is also a major concern. Fewer calculations in a model means less power consumption and smaller battery required, which is more suitable for continuous monitoring.

Another challenge is brought by abnormal gait. Majority of studies in literature use healthy subjects for their gait detection algorithm, which is limited since patients who need to be treated by FES have pathological gait characterized by abnormal speed and asymmetry that is much different from healthy subjects. Studies that classify both healthy and unhealthy gaits generally find better performance for healthy subjects [12]. Thus, a concern for gait segmentation models is their generalizability, or ability to perform well on less abundant pathological gait data. In summary, key limitations in current studies for gait segmentation include

- Lack of models customized for real time gait detection on small, embedded devices.
- Lack of focus on generalizability of models on pathological gait.

Motivated by these gaps in literature, this paper proposes a novel technique for efficient encoding of gait information for increased neural network parameter efficiency. Performance of small CNNs are improved by implementing embedding mapping at the output layer. Embedding is commonly used in language processing to map words to a lower-dimensional space, where words with similar meaning are close together [13]. Due to the cyclical nature gait, gait phases can similarly be mapped to an embedding space where adjacent phases are close together. This contrasts with one-hot encoded classification labels where each class occupy one dimension, thus any two class has the same Euclidean distance to each other. By training embedding to map gait phases to a circular shape, adjacent phases will have closer distance. For example, foot-flat will be closer to toe-off than mid-swing phase.

The main purpose of using embedding is to improve small models' performance by better representing the relationship between phases with cosine similarity. Another approach that achieves a similar purpose is knowledge distillation [14]. Knowledge distillation uses high temperature in the softmax function for a model to encapsulate more information in its output. This is used to teach a smaller model, which learns from the larger model rather than train from scratch. We compare output embedding to knowledge distillation and shows that output embedding is more effective.

We collected original gait data from 15 healthy subjects and 18 patients with osteoarthrosis (OA). CNN are tuned for optimal phase classification on patients with OA. Efficiency techniques including weight pruning and quantization are used to further reduce model sizes. A combination of output embedding mapping and compression enables the development of an accurate lightweight model that can fit easily into a memory limited microcontroller with low inference latency. Output embedding allows a small CNN to improve performance by 7.2% to reach the performance of a large CNN 47 times its size. Compression reduces its size to 15KB and inference latency to 2.76ms whilst preserving good 8-phase classification accuracy of 84.9% on healthy subjects and 67.0% on subjects with OA.

2. Related Work

Gait Segmentation Methods. Models such as Support Vector Machines [15], Bayesian classifier [16], Hidden Markov Model [17] and neural network based methods [18] have been successfully used to classify gait. In particular, convolutional neural networks (CNN) have been an area of interest for gait segmentation due to their ability to detect translationally-invariant features [11], [19]. [9] used gait phase aware receptive field that required only one IMU sensor to adapt to various movement speeds. [20] designed a CNN to distinguish five phases and showed 97% accuracy for offline evaluation.

Neural Network Compression. There are various ways to compress neural networks, or improve performance of a small model. Common ways to reduce CNN size include network pruning, sparse representation, bits precision and knowledge distillation [21]. A combination of these methods such as pruning then quantization [22], weight pruning and knowledge distillation [23] can reduce size further. For gait segmentation, there is a limited number of studies that customize CNN compression for microcontrollers. [24] used width and depth scaling, and quantization to reduce a 28MB model to 0.5KB with minimal loss of accuracy. However, only two phases were distinguished. [25] used execution time and power consumption to optimize three-layer neural networks with weight sharing.

3. The Output Embedding Framework

The output embedding framework is built upon a CNN. A trainable embedding is used to map the output of a naive classification model to a lower dimension that better represents the circular relation between gait phases.

3.1. CNN for Gait Classification

A CNN model was designed to classify gait phase based on 12-dimensional signal from dual thigh-mounted IMUs. To apply 2D convolution, the input was built up by collating samples extending T ms into the past, skipping every ith sample such that the resulting input matrix was of dimensions $[T/i, 12]$. The corresponding desired output was in each case given by the status of the gait phase at time 0; in other words, by the most recent observed value of gait phase. The values of i and T were determined by random search within hyperparameter bounds established through preliminary trail-and-error experimentation.

3.2. Output Embedding

Embedding is the process of redefining vectors in a different vector space, typically of reduced dimensionality. It is commonly used in language processing where words with similar meanings are placed close together in a dense low dimension space rather than using high dimension sparse representation [13]. Here, we explored the use of an embedding mapping to improve efficiency through the encoding of an inductive prior regarding the cyclical nature of gait. In the baseline formulation of the CNN model, the output gait phase is encoded, as per convention, using one-hot encoding, where each of the K phases of gait is represented by a sparse vector of length K and the set of K vectors together represents an orthogonal basis. This representation is inefficient on account of the sparsity and, moreover, on account of orthogonality of the gait phase labels, precludes information on relative closeness of individual phases.

Gait phases are cyclical, so in theory it only takes two dimensions to represent this property. As seen in Figure 6, similarity of two phases can be represented by how close they are on a 2-dimensional circle. However, if K-dimensional space is used to represent each phase as a one-hot encoded vector, Euclidean distance between any two one-hot encoded phases is the same, which fails to detect this intrinsic 2-dimensionality. An trainable embedding can be created to map each phase to a lower dimensional space $N < K$. Let the reduced output be N dimensions. In the output layer of CNNs, the dimension is reduced from K to N. Each output node is given by

$$a_j = \sum_i w_{i,j} x_i \tag{1}$$

where $j = 1, ..., N$ and i is the dimension of the second last layer. The trainable embedding $\boldsymbol{E} \in \mathbb{R}^{K \times N}$ encodes each one hot encoded ground truth labels $\boldsymbol{l}_k \in \mathbb{R}^{1 \times K}, k \in 1, ..., K$ to a lower dimension $\boldsymbol{b}_k = \boldsymbol{l}_k \boldsymbol{E}_k \in \mathbb{R}^{1 \times N}$. Next, the predicted phase $\hat{k}$ is given by

$$\underset{k}{\operatorname{argmax}} f(k) = \frac{\boldsymbol{a} \cdot \boldsymbol{b}_k}{|\boldsymbol{a}||\boldsymbol{b}_k|} \tag{2}$$

Lastly, loss is calculated for back propagation to update the CNN and the embedding using

$$L = CrossEntropy(\boldsymbol{l}_{\hat{k}}, f(\hat{k})) \tag{3}$$

In a perfect model, the output of CNN would overlap with the embedding that represent the correct phase with similarity 1 and loss 0.

3.3. Comparison with Knowledge Distillation

The main purpose of using output embedding is to improve a small model's performance by better representing the relationship between phases. We compare output embedding to knowledge distillation [14] which similarly improves a small model's performance by training it with the knowledge from a larger better performing model. Relationship between phases is better represented by large temperature in the softmax function. Traditionally, the softmax output of an accurate neural network has high probability for the true class, whereas all other classes have low probability. If the probability distribution is more spread out - the loss function shows the degree of similarity between the correct class and other likely classes - then student can learn additional information other than the ground truth label. This is especially pertinent for gait segmentation since gait is cyclical. The fact that adjacent phases are more similar is important information for the student. The probability is softened by temperature T. Given the logits z, each class probability is

$$p_i = \frac{e^{z_i/T}}{\sum_j e^{z_j/T}} \tag{4}$$

Distillation loss is calculated by the weighted sum of the student's Cross Entropy (CE) loss with the ground truth labels, and Kullback-Leibler (KL) divergence with the teacher's soft predictions

$$L_{KD} = \alpha L_{CE}(y, \sigma(z_s)) + (1 - \alpha) L_{KL}(\sigma(z_t; T), \sigma(z_s, T)) \tag{5}$$

where y is the ground truth label, α is the proportion of student loss, σ is the softmax function parameterized by temperature.

4. Experimenets

4.1. Data Collection and Pre-processing

Data was collected using custom hardware incorporating a cortex M4-based microcontroller (STM32H743ZI2, STM) with 3-axis MEMS gyroscopes (FXAS21002, NXP) and 3-axis accelerometer (FXOS8700, NXP). One IMU comprising accelerometer and gyroscope was symmetrically placed mid-thigh bilaterally on each subject's legs. Thus, at each sample time, 12 features were gathered in total – acceleration and angular acceleration in x, y and z directions of each thigh, at a sampling rate of 500Hz. Data were collected from 15 healthy subjects and 18 subjects with osteoarthritis of the knee (OA). Each subject walked freely on a flat surface for 1-5 minutes for each session. Maximum ten sessions were collected for each subject. OA subjects generally walked for shorter durations in each session but participated in more sessions. All subjects signed written consent to have their data collected and the study gained full ethical approval following review by the NHS Research Ethics Committee[1].

Before IMU data is fed to the CNN for training, they are down-sampled by a factor of 10 to be 50 Hz, then normalized by subtracting mean and dividing by standard deviation for each subject. Two independent datasets were available, each comprising the signals from both IMUs but differing in the predicted gait phase variable, with one dataset comprising the phases of the left leg, and one the synchronously recorded phases of the right leg. After axis transformation of the latter dataset for equivalence with the former, the two were concatenated to form a single dataset, thereby obviating the need to train separate models for left and right legs.

A gait cycle of each leg is defined as the period beginning at heel strike, through heel-off and toe-off, and ending at the next heel strike. Within the cycle, stance phase is defined as the period between heel strike and toe off, and

[1] https://www.hra.nhs.uk/about-us/committees-and-services/res-and-recs/

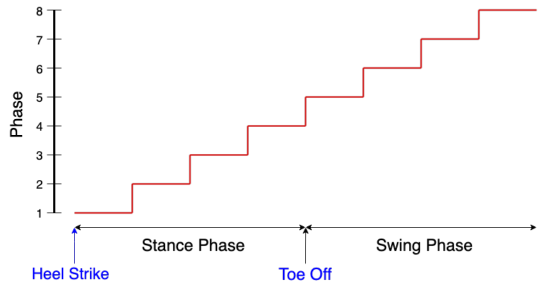


Figure 1: Each gait cycle is divided into 8 phases. Stance and swing phase periods are determined by pressure sensors in subjects' shoes.

swing phase as the period between toe off and subsequent heel strike. In this study, each gait cycle was split into 8 phases based on stance and swing phase. Ground truth labels were derived from stance and swing phase generated by foot pressure sensors. These sensors were placed in foot pads in shoes to provide a binary signal defining swing phase and stance phase, with the former indicated by ongoing pressure above threshold and the latter by pressure below. Each stance and swing phase was divided equally into 4 parts, with phases 1 to 4 representing stance phase and phases 5 to 8 representing swing phase. Thus 8 total phases represented a complete gait cycle as seen in Figure 1.

4.2. CNN Architecture

A baseline CNN model was developed to establish performance bounds for the gait phase classification problem. The baseline model consisted of two main components, convolutional layers and dense layers as seen in Table 1. To apply 2D convolution, samples are downsampled by $i = 5$ times. Past $T = 160\text{ms}$ samples are collated. There are three convolutional layers of increasing sizes with kernels of size 1×5 with no padding and ReLU activation. In order to control the size of activations and therefore memory footprint, each layer is followed by an average-pooling layer that down-samples each output row by taking moving average values of two consecutive values. After the third convolutional layer, a flatten layer converts output to 1D before passing into dense layers. Two densely connected layers with ReLU activation is followed by a dropout layer with dropout probability 0.5. The model is trained until validation accuracy stops increasing, with batch size 128, learning rate 0.001, Adam optimizer, and cross-entropy loss. Since this model exceeds the memory capacity of the STM32H743ZI2 microcontroller used, two other smaller models (CNN-Mid and CNN-Small) were tested. The sizes of individual layers within these models were reduced to limit memory footprint and the number of multiply-accumulate (MAC) operations [26] per inference, so that throughput speed increases. As with the baseline model, hyperparameters were determined by random search within boundaries set by provisional experimentation.

TABLE 1: Architecture of models tested without embedding.

	CNN-Baseline	CNN-Mid	CNN-Small
N1 (conv)	64	16	8
N2 (conv)	128	32	16
N3 (conv)	256	64	32
N4 (dense)	512	128	NA
N5 (dense)	256	64	32
N6 (dense)	Dropout	NA	NA
Output	8	8	8
Complexity (MAC)	1780k	135k	38k
Inference Speed	Out of memory	1500Hz	3000Hz

4.3. Comparison Studies

Mock embedding. To find the best embedding dimension N, values between 2 and 7 are tested. Note that the trainable embedding adds $K \times N$ parameters. One might argue that improved performance might be due to the additional number of parameters. We show that this is not the case by adding the same number of parameters to the CNN output layer. Instead of having a separate embedding mechanism, the the baseline CNNs output layer dimension is changed to N and an additional layer is added after of dimension K so that the the model performs the same classification task. We call this model CNN-MockEmb and compares it with CNN-Emb.

Knowledge distillation. To further evaluate the impact and utility of output embedding, its performance was compared to that of a generic, architecture-agnostic technique for improved performance and efficiency, knowledge distillation [14]. Optimum values for the α and T hyperparamters were determined through random search. We found that knowledge distillation with $\alpha = 0.1$ and T in the range of 5 to 9 improve performance of smaller model most significantly. In following tests, $\alpha = 0.1$ and $T = 9$ are used. Models trained throught knowledge distillation are denoted CNN-KL.

4.4. Compression

Weight pruning and quantization were used to reduced model size further. Fully connected neural networks contain large number of input and output connections between layers. The number of weights is proportional to the storage and computational requirements of the model [27]. By removing unimportant weights below a certain threshold, network complexity can be reduced. Networks are also less prone to over-fitting [28]. After pruning, the weights that are set to zero are still present. Therefore, we use a standard compression algorithm to remove redundancy caused by zero weights. Baseline CNN models has precision 32-bit floating point. We also apply 8-bit quantization so that fewer number of bits is required to represent each weight and the model can be used in an 8-bit microcontroller.

4.5. Evaluation Metric

Asymmetry measures the difference between healthy and pathological gait. A modification of the Robinson Index [29] is used.

$$Asymmetry = 100\frac{T_{left} - T_{right}}{\max(T_{left}, T_{right})} \tag{6}$$

where T is the duration of stance phase. Perfectly symmetrical gait has an index of 0 and the more asymmetrical the gait, the larger the absolute value of the index would be. OA subjects are expected to have more asymmetrical gait.

Testing accuracy criteria consists of three parts.

Absolute Accuracy is the proportion of model prediction phase that agree with ground truth phase for each testing time stamp.

Transition Latency is the absolute value of the time difference between predicted phase transition, and ground truth transition for each pair of adjacent phases. Transition occurs when a series of predictions or labels for a certain phase changes to the next. Transition Latency measures how out of sync the model prediction is from the labels.

Transition Detection Accuracy measures out of all labelled phase transitions, how many are actually detected. This is necessary due to missing phases in certain cases. For OA patients, due to pathological gait, the model fails to detect certain phases but skips over them. This is undesirable because in practice, failure to detect certain phases means a potential for missed intervention.

5. Results and Evaluation

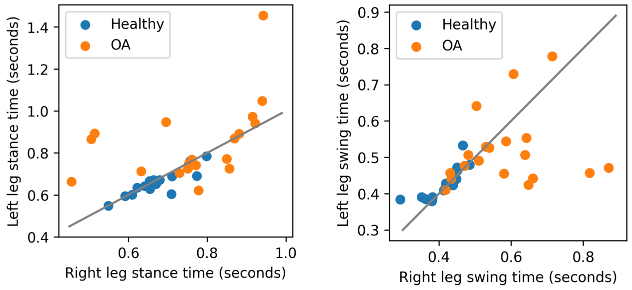
We first describe the gait data collected and the performance of models without embedding. We then demonstrate the effectiveness of output embedding and compare it with knowledge distillation. Lastly, we compress the models to achieve optimal performance.

5.1. Original Data Collected

Data from 15 healthy and 18 patients with OA is visualized in Figure 2. Healthy subjects have stance time lasting 0.55 to 0.8 seconds and swing time lasting 0.3 to 0.5 seconds. Healthy subjects generally walk faster, with shorter stance and swing time. Their gait is also more stable than OA subjects, with smaller standard deviation. OA subjects have more asymmetrical gait where there is a large difference of stance time between left and right leg (Figure 3). Prediction accuracy is correlated with asymmetry where healthy subject's more symmetrical gait has higher accuracy.

5.2. Baseline Model Results

Table 2 shows performance of the baseline model for healthy and OA subjects. Healthy subjects have higher accuracy and lower latency than OA subjects. Healthy subject's phase transition latency is mostly under 20ms. Transition accuracy is close to 1, meaning almost all transitions have been detected. Lower Absolute Accuracy generally correlates with higher Transition Latency and lower Transition Detection Accuracy. In OA patients, average Absolute Accuracy is 67.71% and latency is 53.77ms. The lowest latency is achieved for phase 4 to phase 5 transition, which is expected since this is the toe off action. Interestingly, heel strike action, which is phase 8 to phase 1 transition incurs high latency for OA subjects, but low latency for healthy subjects.



(a) Stance time

(b) Swing time

Figure 2: Left vs right leg swing time and stance time. Healthy subjects generally have more symmetrical gait and shorter gait cycles.

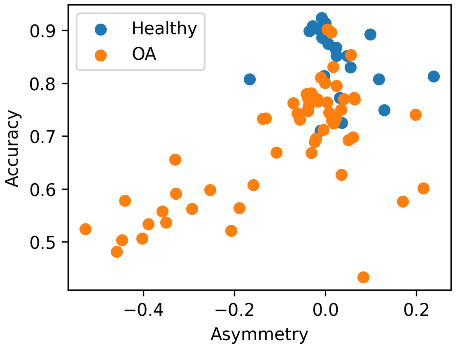


Figure 3: Subjects with OA have more asymmetrical gait.

TABLE 2: Baseline CNN achieves good accuracy and latency

	Healthy subject		OA subject	
Phase Transition	Transition Latency (ms)	Transition Detection Accuracy	Transition Latency (ms)	Transition Detection Accuracy
1→2	16.70	0.963	59.01	0.935
2→3	18.46	0.965	53.23	0.941
3→4	20.83	0.980	41.89	0.955
4→5	17.48	0.970	21.94	0.955
5→6	15.93	0.988	39.04	0.948
6→7	19.97	0.982	64.15	0.927
7→8	19.49	0.974	72.72	0.933
8→1	16.35	0.962	78.17	0.912
Mean	18.15	0.973	53.77	0.938

TABLE 3: Addition of embedding mapping $N = 5$ successfully improves model performance

	Healthy subject			OA subject		
	Absolute Accuracy	Transition Latency (ms)	Transition Detection Accuracy	Absolute Accuracy	Transition Latency (ms)	Transition Detection Accuracy
CNN-Baseline	0.856	18.15	0.973	0.677	53.76	0.938
CNN-Baseline-Emb	0.855	18.41	0.972	0.675	53.18	0.945
CNN-Mid	0.815	19.07	0.880	0.641	55.09	0.853
CNN-Mid-KL	0.845	19.59	0.981	0.670	53.42	0.948
CNN-Mid-Emb	0.853	18.49	0.972	0.674	52.88	0.947
CNN-Small	0.763	18.93	0.752	0.599	56.02	0.715
CNN-Small-KL	0.798	18.39	0.833	0.636	53.95	0.804
CNN-Small-Emb	0.849	19.22	0.980	0.671	55.71	0.944

Although the baseline model performs well, its running memory is too big to fit on a STM32H743ZI2 microcontroller. Thus, two smaller models are proposed. CNN-Mid runs at 1500 Hz which is fast enough while having enough capacity to have decent performance with 3.6% decrease in Absolute Accuracy and 8.5% decrease in Transition Detection Accuracy for OA subjects compared to the baseline. CNN-Small is very fast at 3000 Hz. However, it performs much worse than the baseline where Transition Detection Accuracy drops below 80% (Table 3).

5.3. Effect of Output Embedding

Embedding is used to improve performance of CNN-Mid and CNN-Small up to the baseline level (Table 3). Figure 4 shows that embedding dimension $N = 5$ results in the most significant improvement. Absolute Accuracy is improved to be within 1% of the baseline for both healthy and OA subjects. There is also significant increase in Transition Detection Accuracy, exceeding that of the baseline. Transition Detection Accuracy is much lower for OA subject due to asymmetrical gait.

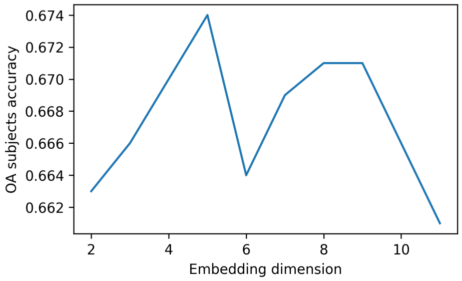


Figure 4: For CNN-Mid, embedding dimension $N = 5$ provides the most improvement for OA subjects.

As seen in Figure 5, for certain OA subjects, CNN-Small is not able to identify all phase transitions. Without embedding, predictions skip over phase 3 for some cycles due to severe asymmetry (-0.33.) This subject tends to favour his right leg during walking. The subject's left leg stance phase is so short that the model cannot detect mid stance. Embedding improves this problem by identifying phases 3 and 4 between phases 1 and 5, even though timing wise they are not perfectly lined up with the ground truth timings. Thus, Transition Detection Accuracy improves significantly compared to without embedding.

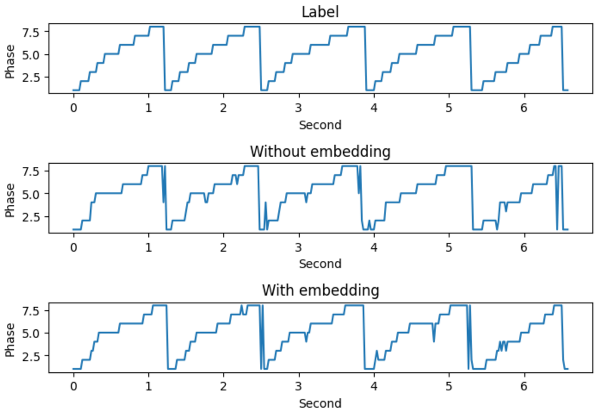
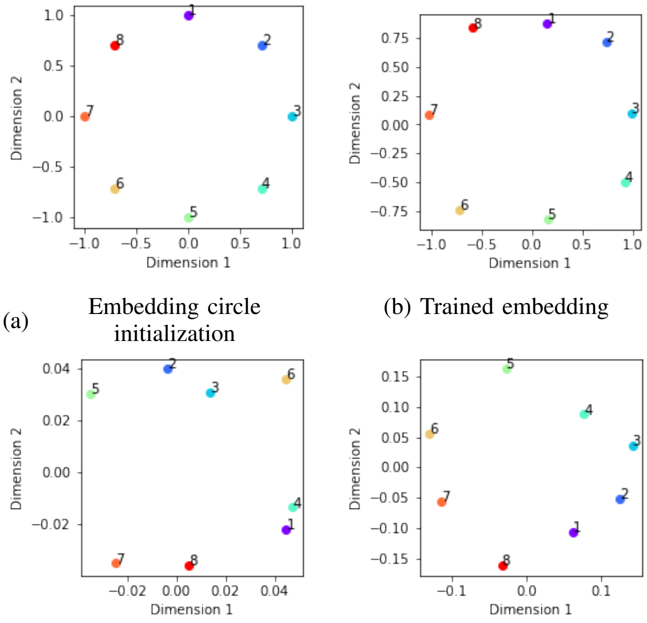


Figure 5: Example of CNN-Small's prediction for a particular subject with OA. Without embedding, the model is not able to identify all phases of gait cycles due to severe asymmetry. Embedding allows the model to distinguish all phases.

Embedding with $N = 5$ is implemented for all three models. CNN-Baseline's performance remains the same with embedding layer. This could be that the baseline model is already tuned to be as accurate as possible, that changing the output to be in an embedding space do not yield additional performance increase. Table 3 shows that for CNN-Mid, large improvements are seen in Absolute Accuracy, Transition Latency and Transition Detection Accuracy, achieving near-parity with the baseline model. For CNN-Small, each metric remains slightly worse than baseline mode, but improves significantly than without embedding. This drastic improvement is explained by improvement in Transition Detection Accuracy. We note that knowledge distillation improves performance of both small models, but not to the same extent as embedding. This further demonstrates the effectiveness of using output embedding.

There are various ways to initialize the embedding layer. An intuitive way is to start embedding as a circle since gait is cyclical. For ease of visualization, 2-dimensional embedding is shown in Figure 6. Embedding was initialized as a circle or randomly. In both cases embedding values were trainable, enabling the model to learn to optimally shift relative vector positionings for increased performance. After training, the embedding shifts to a circular shape, which shows that the model indeed recognizes that gait is circular, and a circular distribution of embedding allows minimization of loss.



(a) Embedding circle initialization

(b) Trained embedding

(c) Embedding random initialization

(d) Trained embedding

Figure 6: Trained 2D embedding always forms a circle, regardless of initialization shape, showing that the model can recognize the circular nature of gait.

It might be proposed that the benefit of embedding is caused by the increase in the number of trainable parameters. This small increase in size is non-negligible for a small model like CNN-Small with a significantly limited size. Hence, a mock model with an extra dense layer added before the output layer is created so that its size is identical to that with embedding. As seen in Table 4, although increasing the size of models by the same amount as embedding does achieve slight improvement in performance, the improvement is no where as significant as embedding. This proves that output embedding mapping has a larger effect of improving performance than simply increasing model size by the same amount.

TABLE 4: Mock embedding simply adds one more layer to CNNs with the same number of additional parameters as trained embedding mapping. Mock embedding underperforms embedding mapping.

	Absolute Accuracy	
	Healthy Subjects	OA Subjects
CNN-Mid-Emb	0.853	0.674
CNN-Mid-MockEmb	0.833	0.649
CNN-Small-Emb	0.849	0.671
CNN-Small-MockEmb	0.754	0.599

5.4. Compression

Lastly, 8-bit quantization and weight pruning are used to effectively decrease model size further (Table 5). Smaller models are pruned to less extent than big model to preserve performance. Thus, CNN-Mid and CNN-Small are compressed by a smaller extent than CNN-Baseline. After compression, CNN-Mid and CNN-Small achieves more than 40 times increase in inference speed with minimal loss in accuracy.

TABLE 5: Each model with embedding is pruned and quantized with minimal loss in accuracy. Absolute Accuracy shows performance on OA patients. Inference time is theoretically calculated from the number of MAC operations.

	Absolute Accuracy	Size (KB)	Inference Time (ms)
CNN-Baseline	0.677	1754	343
+ Compression	0.676	128	18.4
CNN-Mid-Emb	0.674	67	7.75
+ Compression	0.674	24	6.43
CNN-Small-Emb	0.671	19	2.76
+ Compression	0.670	15	2.76

6. Conclusion

In this paper, we collected in house data on healthy and OA patients, and found that highly asymmetrical gait on OA patients results in much lower gait classification accuracy. We designed a large CNN that achieves good accuracy but cannot fit the memory requirement of a microcontroller. Smaller models result in undesirable reduced accuracy. Output embedding mapping is used to improve the results of smaller models by calculating loss using distance based cosine similarity in a reduced dimension, which better represent the relationship between phases. This allows smaller models to reach the accuracy of larger model with minimal addition of trainable parameters. Comparison with knowledge distillation also shows that output embedding is a more effective method to improve the performance of small models. We further compress models with weight pruning and quantization that made models even smaller and achieves minimal latency. The combination of output embedding and compression methods allows smaller models to achieve the same performance as much larger models while being much faster, which is promising for downstream applications on FES that requires precise therapy.

References

[1] G. Lyons, T. Sinkjaer, J. Burridge, and D. Wilcox, “A review of portable fes-based neural orthoses for the correction of drop foot,” *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 10, no. 4, pp. 260–279, 2002.

[2] G. Alon, “Functional electrical stimulation (fes): Clinical successes and failures to date,” *Journal of Novel Physiotherapy and Rehabilitation*, vol. 2, pp. 080–086, 11 2018.

[3] F. B. Wagner, J.-B. Mignardot, C. G. Le Goff-Mignardot, R. Demesmaeker, S. Komi, M. Capogrosso, A. Rowald, I. Seáñez, M. Caban, E. Pirondini, M. Vat, L. A. McCracken, R. Heimgartner, I. Fodor, A. Watrin, P. Seguin, E. Paoles, K. Van Den Keybus, G. Eberle, B. Schurch, E. Pralong, F. Becce, J. Prior, N. Buse, R. Buschman, E. Neufeld, N. Kuster, S. Carda, J. von Zitzewitz, V. Delattre, T. Denison, H. Lambert, K. Minassian, J. Bloch, and G. Courtine, “Targeted neurotechnology restores walking in humans with spinal cord injury,” *Nature*, vol. 563, no. 7729, p. 65—71, November 2018. [Online]. Available: https://doi.org/10.1038/s41586-018-0649-2

[4] D. Laidig, A. J. Jocham, B. Guggenberger, K. Adamer, M. Fischer, and T. Seel, “Calibration-free gait assessment by foot-worn inertial sensors,” *Frontiers in Digital Health*, vol. 3, 2021. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fdgth.2021.736418

[5] A. D. Likens and N. Stergiou, “Chapter 2 - basic biomechanics,” in *Biomechanics and Gait Analysis*, N. Stergiou, Ed. Academic Press, 2020, pp. 17–63. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128133729000026

[6] J. Rueterbories, E. G. Spaich, and O. K. Andersen, “Gait event detection for use in fes rehabilitation by radial and tangential foot accelerations,” *Medical Engineering Physics*, vol. 36, no. 4, pp. 502–508, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1350453313002257

[7] L. Chaikho, E. Clark, and M. Raison, “Transcutaneous functional electrical stimulation controlled by a system of sensors for the lower limbs: A systematic review,” *Sensors*, vol. 22, no. 24, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/24/9812

[8] H. Huang, P. Zhou, Y. Li, and F. Sun, “A lightweight attention-based cnn model for efficient gait recognition with wearable imu sensors,” *Sensors*, vol. 21, no. 8, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/8/2866

[9] J.-D. Sui, W.-H. Chen, T.-Y. Shiang, and T.-S. Chang, “Real-time wearable gait phase segmentation for running and walking,” in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020, pp. 1–5.

[10] R. Evans and D. Arvind, “Detection of gait phases using orient specks for mobile clinical gait analysis,” in *2014 11th International Conference on Wearable and Implantable Body Sensor Networks*, 2014, pp. 149–154.

[11] T. Zhen, L. Yan, and P. Yuan, “Walking gait phase detection based on acceleration signals using lstm-dnn algorithm,” *Algorithms*, vol. 12, no. 12, 2019. [Online]. Available: https://www.mdpi.com/1999-4893/12/12/253

[12] D. Sethi, S. Bharti, and C. Prakash, “A comprehensive survey on gait analysis: History, parameters, approaches, pose estimation, and future work,” *Artificial Intelligence in Medicine*, vol. 129, p. 102314, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0933365722000793

[13] F. Almeida and G. Xexéo, “Word embeddings: A survey,” 2023.

[14] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” 2015.

[15] R. Begg, M. Palaniswami, and B. Owen, “Support vector machines for automated gait classification,” *IEEE transactions on bio-medical engineering*, vol. 52, pp. 828–38, 06 2005.

[16] H. Manap, N. Tahir, and R. Abdullah, “Anomalous gait detection using naive bayes classifier,” 09 2012, pp. 378–381.

[17] L. Liu, H. Wang, H. Li, J. Liu, S. Qiu, H. Zhao, and X. Guo, “Ambulatory human gait phase detection using wearable inertial sensors and hidden markov model,” *Sensors*, vol. 21, no. 4, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/4/1347

[18] Q. Zou, Y. Wang, Q. Wang, Y. Zhao, and Q. Li, “Deep learning-based gait recognition using smartphones in the wild,” 2020.

[19] D.-X. Liu, X. Wu, W. Du, C. Wang, and T. Xu, “Gait phase recognition for lower-limb exoskeleton with only joint angular sensors,” *Sensors*, vol. 16, no. 10, p. 1579, 2016.

[20] B. Su, C. Smith, and E. Gutierrez Farewik, “Gait phase recognition using deep convolutional neural network with inertial measurement units,” *Biosensors*, vol. 10, no. 9, p. 109, 2020.

[21] R. Mishra, H. P. Gupta, and T. Dutta, “A survey on deep neural network compression: Challenges, overview, and solutions,” 2020.

[22] K. Wróbel, M. Pietroń, M. Wielgosz, M. Karwatowski, and K. Wiatr, “Convolutional neural network compression for natural language processing,” 2018.

[23] N. Aghli and E. Ribeiro, “Combining weight pruning and knowledge distillation for cnn compression,” in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021, pp. 3185–3192.

[24] Y.-A. Chen, J.-D. Sui, and T.-S. Chang, “Real time on sensor gait phase detection with 0.5kb deep learning model,” 2022.

[25] F. Luna-Perejón, M. Domínguez-Morales, D. Gutiérrez-Galán, and A. Civit-Balcells, “Low-power embedded system for gait classification using neural networks,” *Journal of Low Power Electronics and Applications*, vol. 10, no. 2, 2020. [Online]. Available: https://www.mdpi.com/2079-9268/10/2/14

[26] M. Taghavi and M. Shoaran, “Hardware complexity analysis of deep neural networks and decision tree ensembles for real-time neural data classification,” 03 2019, pp. 407–410.

[27] R. Mishra, H. P. Gupta, and T. Dutta, “A road health monitoring system using sensors in optimal deep neural network,” *IEEE Sensors Journal*, vol. 21, no. 14, pp. 15 527–15 534, 2021.

[28] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, “Rethinking the value of network pruning,” 2019.

[29] S. Viteckova, P. Kutilek, Z. Svoboda, R. Krupicka, J. Kauler, and Z. Szabo, “Gait symmetry measures: A review of current and prospective methods,” *Biomedical Signal Processing and Control*, vol. 42, pp. 89–100, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1746809418300193