Lightweight Non-Intrusive Load Monitoring Employing Pruned Sequence-to-Point Learning

Jack Barber, Heriberto Cuayáhuitl
School of Computer Science,
University of Lincoln, UK
17633953@students.lincoln.ac.uk
hcuayahuitl@lincoln.ac.uk

Mingjun Zhong
Department of Computing Science,
University of Aberdeen
Aberdeen, United Kingdom
mingjun.zhong@abdn.ac.uk

Wenpeng Luan
College of Electrical and Information
Engineering, Tianjin University
Tianjin, China
wenpeng.luan@tju.edu.cn

Abstract

Non-intrusive load monitoring (NILM) is the process in which a household's total power consumption is used to determine the power consumption of household appliances. Previous work has shown that sequence-to-point (seq2point) learning is one of the most promising methods for tackling NILM. This process uses a sequence of aggregate power data to map a target appliance's power consumption at the midpoint of that window of power data. However, models produced using this method contain upwards of thirty million weights, meaning that the models require large volumes of resources to perform disaggregation. This paper addresses this problem by pruning the weights learned by such a model, which results in a lightweight NILM algorithm for the purpose of being deployed on mobile devices such as smart meters. The pruned seq2point learning algorithm was applied to the REFIT data, experimentally showing that the performance was retained comparing to the original seq2point learning whilst the number of weights was reduced by 87%. Code:https://github.com/JackBarber98/pruned-nilm

CCS Concepts • **Computing methodologies** → *Supervised learning by regression*; **Neural networks**.

Keywords Energy disaggregation, deep learning

ACM Reference Format:
Jack Barber, Heriberto Cuayáhuitl, Mingjun Zhong, and Wenpeng Luan. 2020. Lightweight Non-Intrusive Load Monitoring Employing Pruned Sequence-to-Point Learning. In *NILM 2020: 5th International Workshop on Non-Intrusive Load Monitoring, November 18, 2020, Virtual*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/1122445.1122456

5th International Workshop on Non-Intrusive Load Monitoring, November 18, 2020, Online

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

1 Introduction

The purpose of NILM is to break down the household's total power consumption into individual appliance power levels. When NILM is applied, the smart meter could provide feedback to the users informing the power consumption of appliances and consequently help householders to reduce the energy consumption. Research has shown that NILM could help household users to reduce energy consumption by 15% [12]. Additionally, smart meters integrated with NILM could help energy providers to optimise their smart grid operations and as well as propose specific tariffs for users according to their energy consumption habits, e.g. NILM could be used to learn the appliance usage patterns [14].

Contemporary solutions using deep neural nets (DNNs) are the state-of-the-art approaches to NILM, arguably comparing to other statistical and signal processing methods [4, 5, 7, 17, 30–33]. Various DNN architectures have been investigated for NILM, which include convolutional neural networks (CNNs) [3, 8, 10, 11, 18, 23, 29], recurrent neural networks (RNNs) [18, 21], denoising autoencoders [6, 18], generative adversarial networks [2], residual attention networks [28], dilated networks [13, 16, 28], and gated networks [25]. These learning algorithms require large amounts of time and hardware resources when training and making inferences. This limitation currently makes domestic energy disaggregation systems for home smart meters largely unavailable. For example, the seq2point learning model contains approximately thirty million weights. Techniques are required to minimise this number as much as possible to deliver a mobile energy disaggregation system. For mitigating this limitation, this paper proposes to inspect pruning the network to minimise both the size and inference time of such neural network algorithms. Four pruning techniques are studied: structured probabilistic pruning [27], entropy-based pruning [15], relative threshold pruning [1], and constant sparsity low magnitude pruning [26]. The purposes of these pruning techniques are to remove any weights from the networks that do not contribute significantly to the networks' output and thus do not affect the performance of the algorithm.

In addition, the architecture of seq2point learning could be made lighter by reducing the number of filters of CNNs and as well as the number of neurons. This paper applies the seq2point learning algorithm with pruning and a reduced architecture to a real-world household energy data. Experimental results show that the performance of the pruned and reduced model was similar to the original model whilst the number of weights was reduced by 87%.

2 Sequence-to-Point Learning

The sequence-to-point model was proposed in [29] for use in the context of NILM. It slides a window over the electricity main readings in a household, which is used as the input to a CNN with five layers followed by two fully connected layers (see the architecture). Correspondingly, a single midpoint at the centre of the window of an appliance is used as the target of the network. A similar scheme has been used for the spectral analysis of speech. For example, seq2point models have been used to infer a single value from a sequence of time ordered data [24].

- Seq2point Architecture [29]
  - Input sequence with length W: $Y_{t:t+\quad 1}$
  - 1D Convolution: {# filters: 30; filter size: 10}
  - 1D Convolution: {# filters: 30; filter size: 8}
  - 1D Convolution: {# filters: 40; filter size: 6}
  - 1D Convolution: {# filters: 50; filter size: 5}
  - 1D Convolution: {# filters: 50; filter size: 5}
  - Fully connected: {# units: 1024}
  - Output: {Number of units:1, activation: linear}

Recent deep learning approaches to NILM have been largely developed over the seq2point approach, showing that it would be one of the most promising approaches to ultimate NILM. Most of previous works have attempted to improve the performance of seq2point by using various variants of architectures including gated CNN and RNNs, very deep CNNs, fast sequence-to-point learning with CNN/RNN/WaveNet, and dilated residual attention networks [8, 9, 16, 21, 23, 25, 28]. These models require an enormous number of weights to represent the networks and thus need large amounts of memory for deployment. Little work has been devoted to lightweight nets for NILM, which require less parameters and memory for the purpose of deployment on mobile devices. In the following section, methods including reduced and pruned networks are proposed as preliminary approaches to lightweight NILM. Although this architecture was used in this paper, these pruning algorithms could also be applied to other DNN models.

3 Pruning Methods

There are two approaches to reduce the number of weights in seq2point learning whilst preserving their performance. Whilst the first approach uses dropout and a reduced number of CNN filters to reduce the size of the networks, the second uses pruning methods to prune the learned weights.

3.1 Reducing Network Architectures

3.1.1 Dropout. One aspect of the seq2point model that has potential to be improved is the structure of the net itself. Dropout has traditionally been used to enhance the generalisability of an ANN, but as dropout randomly removes (drops) a specified percentage of weights from a neural network, it has the side-effect of increasing a model's sparsity by a controllable amount. We therefore applied dropout into the seq2point architecture.

3.1.2 Reduced Architecture. One the easiest ways to reduce the number of neurons and weights in a seq2point architecture is to reduce the network size by making its layers smaller. Naturally, a CNN that contains too few convolutional filters or dense layer weights will never be able to make accurate predictions or classifications – a balance must be found between performance and model size. There are no rules and guidelines to determine the optimal number of neurons and weights. The required number depends upon parameters such as the data being used and the task a network is being trained to complete. We propose to reduce the number of filters in each convolutional layer by ten, and also to reduce the number of neurons in the net's dense layer from 2^{10} to 2^9 neurons. In the following we investigate pruning algorithms to further reduce the number of weights.

3.2 Pruning Network Weights

3.2.1 Structured Probabilistic Pruning . Pruning weights of networks is to shrink the number of weights in order to reduce the size of the network whilst their performance is preserved. For this purpose, unlike other pruning algorithms—typically designed for CNNs—the structured probabilistic pruning (SPP) algorithm [27] groups weights by filters instead of by layers. This means that 2D convolutional layers like those seen in the seq2point model are less prone to being heavily pruned compared to fully connected layers. Further details of this method can be found in [27].

3.2.2 Entropy-Based Pruning. An entropy-based approach to pruning (EBP) [15] uses the information contribution of a weight to determine whether it should be pruned. It aimed to reduce the loss in inference accuracy that occurs when performing sparsification on a network. The method prunes the network layer-wise under the assumption that the distribution of weights in a layer is quasi-Gaussian, see [15].

3.2.3 Relative Threshold Pruning. Relative threshold pruning (RTP) has been proposed by [1], where a threshold value below which pruning occurs is identified and enforced by the algorithms. The best performing algorithm demonstrated determines pruning thresholds for each layer relative to the size of each layer. The value of the threshold, denoted

by τ_l, is set such that τ_l $percentile(\ _l, w_l)$, where $_l$ is a desired percentile of the layer's weights, w_l, to obtain. Thus, the number of weights pruned becomes entirely dependent on the distribution of a layer's weights. $_l$ values for the upper and lower thresholds of a layer must be individually selected, and the thresholds for convolutional layers must have a smaller range to ensure models can still converge.

3.2.4 Constant Sparsity Low Magnitude Pruning. TensorFlow's Model Optimisation library [26] provides a simple low magnitude pruning algorithm. This algorithm removes a fixed, pre-defined proportion of weights with the lowest absolute values from a neural network model during training. Two variants of this algorithm—herein referred to as TFMOT (TensorFlow Model Optimisation Toolkit) pruning—are available for use: (1) polynomial decay low-magnitude pruning, and (2) constant sparsity low-magnitude pruning (LMP). The former reduces the number of weights gradually until the target sparsity is reached, whereas the constant sparsity approach immediately produces a model with the target sparsity and adjusts the weights that have been removed to enhance accuracy during each pruning iteration.

4 Experimental Setup

4.1 Data Sets

The most used datasets in recent NILM-based research include UK-DALE [19], REDD [20], and REFIT [22]. They all allow for the power consumption of the device to be calculated or used directly. The REFIT dataset is larger than others, so it will be utilised in the experiments presented here. This dataset contains data gathered from various appliances from twenty different properties at eight second intervals. The dataset contains 6.46GB of power readings, and therefore should be suitable for producing models generalised to their appliance domain. As training a model for each available target appliance would require considerable computing resources, two appliances will be used: the kettle and dishwasher. These have been selected to demonstrate performance with devices with energy signatures of varying complexities; the kettle can either be ON or OFF, whereas the dishwasher can have a wide range of states and can be operational for highly variable periods of time. A sliding window containing 599 data points, i.e., 4792 seconds, will be used. Note that this number of data points is identical to the original seq2point paper [29].

4.2 Metrics

Two common error metrics will be used to evaluate the performance of our regression models: mean absolute error (MAE), and mean squared error (MSE). As these do not provide any indication of how effective the pruning process was, the number of non-zero weights present in a model will be recorded to produce the sparsity measure S $\frac{len(w\ 0)}{len(w)}$, where w_l is a layer l's weight matrix.

4.3 Network and Algorithm Parameters

The network settings specified by D'Incecco et al. [10] will be used when defining models. Where it is possible to set the target sparsity ratio of a pruning algorithm, it will be set to 0.7 in an effort to ensure that only the process by which weights are chosen for pruning by an algorithm are responsible for a model's relative performance. However, this target sparsity may not be achieved in all instances due to the deployment of early stopping.

5 Results

5.1 Pruning Algorithms

We evaluate the pruning algorithms described in Section 3, and apply them to kettle and dishwasher appliances to choose the best pruning algorithm for seq2point learning.

5.1.1 Kettle. The SPP algorithm produced a model with the lowest errors compared to other models trained to disaggregate kettle energy values. See the results in the Table 1. SPP yielded a MAE 11.6% smaller than LMP which was the next best pruning algorithm and was 9.8% smaller than that of the control model. However, although SPP removed 23% of weights from the model's convolutional layers, the overall sparsity ratio of the model was negligible as SPP only prunes weights from a model's convolutional layers.

The next-best pruning algorithm was LMP with a MAE of 2.0% greater than the control model, and shows that seq2point models can indeed be pruned with a minimal negative impact on inference capability. LMP also has the highest sparsity ratio of all kettle models at 70%, meaning that the number of non-zero weights the model contains was reduced from 30 million to 9.2 million. This is an extremely significant reduction with minimal performance degradation and appears to offer the optimal balance between sparsity ratio and error.

Both MSE and MAE of EBP were similar to that of LMP. However, the model pruned using EBP had a sparsity ratio 39% less than the LMP model. This model still shows potential as a significant level of pruning occurred whilst maintaining good inference ability compared to the control model.

Whilst a high sparsity ratio of 40% was achieved by the model trained using the relative threshold pruning approach, this was the worst-performing model. The significant degradation in MAE and MSE leads to the conclusion that this algorithm is not suitable for pruning seq2point models for use in energy disaggregation.

5.1.2 Dishwasher. None of the pruning algorithms performed better than the control model when models were trained to infer values for dishwashers. While SPP yielded the lowest MSE, the EBP algorithm resulted in the lowest MAE. SPP produced a model with an MSE 4.6% greater than

Table 1. Four pruning methods applied to seq2point model. The LMP achieved the similar performance comparing to the seq2point without pruning (control) with the best sparsity 70%. (Best results are in bold.)

	Control		Low Magnitude Pruning (LMP)		Relative Threshold Pruning (RTP)		Structured Probabilistic Pruning (SPP)		Entropy-Based Pruning (EBP)	
Number of Weights	30,707,024		**9 212 107**		18,424,214		30,636,624		21,187,847	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
Kettle	0.0712	0.1102	0.0726	0.1151	0.0819	0.1269	**0.0642**	**0.0990**	0.0766	0.1150
Dishwasher	0.0878	**0.1642**	0.1011	0.1814	0.1072	0.2033	0.0824	0.1717	**0.0799**	0.1797
Mean	0.0795	0.1372	0.0869	0.1483	0.0946	0.1651	**0.0733**	**0.1354**	0.0783	0.1474

Table 2. Reduced and dropout methods applied to LMP. The LMP with reduced and dropout methods achieved even better performance comparing to seq2point (control) whilst the sparsity achieved 87%.

	LMP		Reduced LMP		Dropout LMP		Reduced + Dropout LMP	
Number of Weights	9,212,107		**3 685 637**		9,212,107		**3 685 637**	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
Kettle	0.0726	0.1151	0.0756	0.1174	0.0745	0.0420	**0.0650**	**0.0335**
Dishwasher	0.1011	0.1814	0.0558	0.0649	0.0862	0.0911	**0.0401**	**0.0487**
Mean	0.0869	0.1483	0.0657	0.0912	0.0804	0.1331	**0.0525**	**0.0411**

the control model's, which is an acceptable increase in error as it less than 5%. The MAE of this model was smaller than the control model's, and whilst the compression ratio yielded by this model was insignificant it should not be disregarded as it may be possible to use this algorithm in conjunction with others to prune a model to a higher standard.

The model trained using the EBP had a MAE 7.9×10^{-4} less than the control model, whilst its MSE was 1.55×10^{-3} greater. This implies that individual errors experiences were more likely to be large in absolute value compared to the control. However, this algorithm caused little net performance loss whilst pruning 23% of weights present in the model. The worst-performing algorithm was once again the RTP. The MAE of the model produced saw an increase in MAE of 22.1% versus the control, whilst the MSE saw a similar increase of 23.8%. The MAE of this algorithm's model is 34.2% greater than that of the model produced via EBP. Regardless of the model's 40% sparsity, this large increase in error makes the RTP unsuitable for energy disaggregation.

LMP was the second-best performing algorithm in terms of sparsity ratio. The dramatic number of weights this algorithm appears to remove from models arguably offsets the fact that it does not cause the least degradation inference performance. As this algorithm produced high-quality models for both the kettle and dishwasher devices, it can be concluded that this is the most suitable pruning algorithm evaluated.

5.2 Reduced and Dropout Models

Results show that LMP was the optimal pruning algorithm used. Thus, a reduced seq2point model with LMP and dropout was trained. The results are shown in the Table 2. The last column of Table 2 shows that when both reduced architecture and dropout along with LMP were used, the MSE produced were only one third of the control model. The MAE of these models were also significantly smaller, suggesting at a surface level that this composition is optimal and importantly the sparsity achieved 87%. Applying just dropout alongside pruning yields models with very small error metrics compared to the control architecture, out-performed only by the models with dropout and a reduced number of weights and filters. Combing reduced and LMP reached sparsity by 87%, but the performance in terms MAE and MSE was not better than LMP with reduced and dropout model.

6 Conclusions

This work demonstrates that seq2point models can be pruned to remove unnecessary weights with minimal performance degradation. LMP was the optimal algorithm among those pruning algorithms, and that this can successfully be applied to a reduced seq2point architecture with dropout. The results show that the LMP with reduced and dropout model outperformed the original seq2point model and as well as reduced the number of weights by 87%. Future research could focus on finding an optimal network architecture that balances performance with network size and applying them to other NILM approaches. It is hoped that this may aid in the development of an oracle energy disaggregation system that is lightweight and capable of performing real-time disaggregation on extremely low-power computing devices such as smart meters.

7 Acknowledgement

WL is supported by The National Key R&D Program of China (No. 2018YFB0904502).

References

[1] Amir H. Ashouri, Tarek S. Abdelrahman, and Alwyn [Dos Remedios]. 2019. Retraining-free methods for fast on-the-fly pruning of convolutional neural networks. *Neurocomputing* 370 (2019). https://doi.org/10.1016/j.neucom.2019.08.063

[2] Kaibin Bao, Kanan Ibrahimov, Martin Wagner, and Hartmut Schmeck. 2018. Enhancing neural non-intrusive load monitoring with generative adversarial networks. *Energy Informatics* 1, 1 (2018).

[3] Nipun Batra, Rithwik Kukunuri, Ayush Pandey, Raktim Malakar, Rajat Kumar, Odysseas Krystalakos, Mingjun Zhong, Paulo Meira, and Oliver Parson. 2019. Towards reproducible state-of-the-art energy disaggregation. In *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*. 193–202.

[4] Nipun Batra, Amarjeet Singh, and Kamin Whitehouse. 2016. Gemello: Creating a Detailed Energy Breakdown from Just the Monthly Electricity Bill. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *KDD '16)*. Association for Computing Machinery, New York, NY, USA, 431–440. https://doi.org/10.1145/2939672.2939735

[5] M. Z. A. Bhotto, S. Makonin, and I. V. Bajić. 2017. Load Disaggregation Based on Aided Linear Integer Programming. *IEEE Transactions on Circuits and Systems II: Express Briefs* 64, 7 (2017), 792–796.

[6] Roberto Bonfigli, Andrea Felicetti, Emanuele Principi, Marco Fagiani, Stefano Squartini, and Francesco Piazza. 2018. Denoising autoencoders for Non-Intrusive Load Monitoring: Improvements and comparative evaluation. *Energy and Buildings* 158 (2018).

[7] Roberto Bonfigli, Stefano Squartini, Marco Fagiani, and Francesco Piazza. 2015. Unsupervised algorithms for non-intrusive load monitoring: An up-to-date overview. In *2015 IEEE 15th International Conference on Environment and Electrical Engineering EEEIC)*. IEEE, 1175–1180.

[8] Kunjin Chen, Qin Wang, Ziyu He, Kunlong Chen, Jun Hu, and Jinliang He. 2018. Convolutional sequence to sequence non-intrusive load monitoring. *The Journal of Engineering* 2018, 17 (2018), 1860–1864.

[9] Prajna Dash and Kshirasagar Naik. 2018. A Very Deep One Dimensional Convolutional Neural Network (VDOCNN) for Appliance Power Signature Classification. In *2018 IEEE Electrical Power and Energy Conference EPEC)*. IEEE, 1–6.

[10] M. D'Incecco, S. Squartini, and M. Zhong. 2020. Transfer Learning for Non-Intrusive Load Monitoring. *IEEE Transactions on Smart Grid* 11, 2 (2020), 1419–1429.

[11] Anthony Faustine and Lucas Pereira. 2020. Improved appliance classification in non-intrusive load monitoring using weighted recurrence graph and convolutional neural networks. *Energies* 13, 13 (2020), 3374.

[12] Corinna Fischer. 2008. Feedback on household electricity consumption: a tool for saving energy? *Energy efficiency* 1, 1 (2008), 79–104.

[13] A. Harell, S. Makonin, and I. V. Bajić. 2019. Wavenilm: A Causal Neural Network for Power Disaggregation from the Complex Power Signal. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP)*. 8335–8339.

[14] Chen-Yu Hsu, Abbas Zeitoun, Guang-He Lee, Dina Katabi, and Tommi Jaakkola. 2020. Self-Supervised Learning of Appliance Usage. In *ICLR*. https://openreview.net/forum?id=B1lJzyStvS

[15] Cheonghwan Hur and Sanggil Kang. 2019. Entropy-based pruning method for convolutional neural networks. *The Journal of Supercomputing* 75:6 (2019).

[16] Jie Jiang, Qiuqiang Kong, Mark Plumbley, and Nigel Gilbert. 2019. Deep Learning Based Energy Disaggregation and On/Off Detection of Household Appliances. *arXiv preprint arXiv:1908.00941* (2019).

[17] Richard Jones, Christoph Klemenjak, Stephen Makonin, and Ivan V Bajic. 2020. Exploring Bayesian Surprise to Prevent Overfitting and to Predict Model Performance in Non-Intrusive Load Monitoring. *arXiv preprint arXiv:2009.07756* (2020).

[18] Jack Kelly and William Knottenbelt. 2015. Neural NILM: Deep neural networks applied to energy disaggregation. In *BuildSys*. ACM.

[19] Jack Kelly and William Knottenbelt. 2015. The UK-DALE dataset, domestic appliance-level electricity demand and whole-house demand from five UK homes. *Scientific Data* 2, 150007 (2015).

[20] J Zico Kolter and Matthew J Johnson. 2011. REDD: A public data set for energy disaggregation research. In *SIGKDD Workshop on Data Mining Applications in Sustainability*, Vol. 25. Citeseer.

[21] Odysseas Krystalakos, Christoforos Nalmpantis, and Dimitris Vrakas. 2018. Sliding window approach for online energy disaggregation using artificial neural networks. In *Hellenic Conference on Artificial Intelligence*.

[22] David Murray, Lina Stankovic, and Vladimir Stankovic. 2017. An electrical load measurements dataset of United Kingdom households from a two-year longitudinal study. *Scientific data* 4 (2017), 160122.

[23] David Murray, Lina Stankovic, Vladimir Stankovic, Srdjan Lulic, and Srdjan Sladojevic. 2018. Transferability of neural networks approaches for low-rate energy disaggregation. In *ICASSP*.

[24] Tara N. Sainath, Brian Kingsbury, George Saon, Hagen Soltau, Abdelrahman Mohamed, George Dahl, and Bhuvana Ramabhadran. 2015. Deep Convolutional Neural Networks for Large-scale Speech Tasks. *Neural Networks* 64 (2015).

[25] Changho Shin, Sunghwan Joo, Jaeryun Yim, Hyoseop Lee, Taesup Moon, and Wonjong Rhee. 2019. Subtask Gated Networks for Non-Intrusive Load Monitoring. *AAAI* (2019).

[26] Tensorflow. 2019. *TensorFlow Model Optimization Toolkit — Pruning API*. https://blog.tensorflow.org/2019/05/tf-model-optimization-toolkit-pruning-API.html

[27] Huan Wang, Qiming Zhang, Yuehai Wang, and Haoji Hu. 2018. Structured Probabilistic Pruning for Convolutional Neural Network Acceleration. In *BMVC*.

[28] Min Xia, Yiqing Xu, Ke Wang, Xu Zhang, et al. 2019. Dilated residual attention network for load disaggregation. *Neural Computing and Applications* 31, 12 (2019).

[29] Chaoyun Zhang, Mingjun Zhong, Zongzuo Wang, Nigel Goddard, and Charles Sutton. 2018. Sequence-to-point learning with neural networks for nonintrusive load monitoring. In *AAAI*.

[30] Bochao Zhao, Minxiang Ye, Lina Stankovic, and Vladimir Stankovic. 2020. Non-intrusive load disaggregation solutions for very low-rate smart meter data. *Applied Energy* 268 (2020), 114949.

[31] Mingjun Zhong, Nigel Goddard, and Charles Sutton. 2014. Interleaved factorial non-homogeneous hidden Markov models for energy disaggregation. *arXiv preprint arXiv:1406.7665* (2014).

[32] Mingjun Zhong, Nigel Goddard, and Charles Sutton. 2014. Signal Aggregate Constraints in Additive Factorial HMMs, with Application to Energy Disaggregation. In *Advances in Neural Information Processing Systems 27*. pp.3590–3598.

[33] Mingjun Zhong, Nigel Goddard, and Charles Sutton. 2015. Latent Bayesian melding for integrating individual and population models. In *Advances in Neural Information Processing Systems 28*. pp.3618–3626.