Learning to smell for wellness

Kehinde Owoeye
Department of Computer Science
University College London
London, WC1E 6EA
ucabowo@ucl.ac.uk

Abstract

Learning to automatically perceive smell is becoming increasingly important with applications in monitoring the quality of food and drinks for healthy living. In todays age of proliferation of internet of things devices, the deployment of electronic nose otherwise known as smell sensors is on the increase for a variety of olfaction applications with the aid of machine learning models. These models are trained to classify food and drink quality into several categories depending on the granularity of interest. However, models trained to smell in one domain rarely perform adequately when used in another domain. In this work, we consider a problem where only few samples are available in the target domain and we are faced with the task of leveraging knowledge from another domain with relatively abundant data to make reliable inference in the target domain. We propose a weakly supervised domain adaptation framework where we demonstrate that by building multiple models in a mixture of supervised and unsupervised framework, we can generalise effectively from one domain to another. We evaluate our approach on several datasets of beef cuts and quality collected across different conditions and environments. We empirically show via several experiments that our approach perform competitively compared to a variety of baselines.

1 Introduction

Safeguarding the health and well being of millions of people in the world most especially in the developing regions of the world remain one of the seventeen key 2030 Agenda for Sustainable Development of the United Nations United Nations [2015]. To achieve this, the quality of food and drink products remains a target to be monitored by appropriate authorities to ensure they are safe and healthy for all and sundry.

Due to the proliferation of Internet of things devices, gas sensors in the form of electronic nose are becoming increasingly available and important for smelling and tasting chemicals, food and wines Rodríguez-Méndez et al. [2016], Wijaya et al. [2018a] for the purpose of assessing their quality. The data obtained from these devices can be used to build a machine learning model with applications in predicting in the future the exact quality of food and drink products at different levels of granularity. However, like most machine learning models, these models on their own do not scale when used in other different but similar domains where the features are different due to covariate shift or when there is a mismatch in the distribution of the labels in the respective domains.

Existing methods proposed towards tackling this problem with respect to time series data have however been designed for domains where either data has been collected in well controlled environments Purushotham et al. [2016], simple binary classification problems Purushotham et al. [2016] or has considered separately the problem of domain adaptation and semi supervised learning in the same domain with few data points Zhu et al. [2018]. There are however problems with these methods in relation to the contexts where they have been used. While weakly supervised learning

Workshop on AI for Social Good workshop NeurIPS (2019), Vancouver, Canada.

methods don't scale to other domains, recent work Saito et al. [2019] has shown that conventional unsupervised domain adaptation methods designed to produce domain invariant features still perform poorly even when few samples are available in the target domain as well as fail to address adequately classification problems that exists around class boundaries in the target domain. In addition, while domain adaptation problems have hugely focused on generating domain invariant features, in practice, there is a mismatch in the label space coupled with the noisy nature of sensor data.

In this paper, we consider a scenario where we address the problem of domain adaptation with only few samples (four samples) Xu and Tenenbaum [2007] also known as semi-supervised/weakly supervised/few shots domain adaptation. Furthermore, we consider a situation where the classification is more fine-grained with the potential to misclassify for a naive classifier. We propose an approach that leverages a hierarchical model to find sub-groups in the source domain in an unsupervised manner using clustering. These sub-groups are then trained separately in a supervised learning framework with the aid of a recurrent neural network. A classifier is further trained on four samples per class in the target domain to map these data to the source domain clusters or models where the probability of classifying them accurately is best maximised in addition to training the source domain data in each cluster with the few labeled target domain data. We evaluate our approach on datasets of beef meat quality of different cuts collected across different spatiotemporal domains. Results on a variety of experiments with these datasets show that our approach performs competitively compared to competing baselines.

2 Related Work

We discuss previous works relative to ours under three broad themes of transfer learning, semi-supervised learning and domain adaptation.

Transfer Learning: Using all 85 datasets in the UCR archive Chen et al. [2015], a convolutional neural network (CNN) was proposed to classify time-series Fawaz et al. [2018]. They concluded that source data with some similarity to the target result in positive transfer and negative transfer if there is no similarity. In our case, we are only interested in using dataset with similarities in this case different beef cuts collected across different conditions but with varying difference in the distribution of the input features and labels.

Semi-Supervised Learning: A lot of work have been carried out in this space with respect to time series Wei and Keogh [2006], Guan et al. [2007]. One key difference in our approach with respect to these works is that these methods are only designed for the domain where the source data is collected and perform poorly outside of this domain when the input and target distribution changes. In addition, we only consider a much more difficult few shot learning scenario where there are not mot more than four samples per class in the target domain.

Domain adaptation: Building on the method proposed by Ganin et al. [2016], a variational recurrent adversarial domain adaptation Purushotham et al. [2016] was proposed to generate domain invariant features for healthcare time series data. Compared to the binary classification problem considered in this work, we have focused on even more challenging task of classifying noisy time-series data into four groups where there are non-trivial differences between the input and label data distribution in the source and target domain. In addition, while they have assumed access to all the input features of the target domain, we assume we only have access to just four samples per class with their associated labels.

3 Problem formulation & Training objectives

3.1 Problem Formulation

Given two time series distributions $S(x_t, y_t)_{t=1}^{N_S}$ and $T(x_t, y_t)_{t=1}^{N_T}$ where the former represents the source domain and the latter the target domain while x_t and y_t represents the input features and the labels at each time-step t respectively. Furthermore, N_S and N_T may or may not be equal and denotes the respective length of the two distributions and also the two distributions are different but similar in some respects. We assume during training we have access to all of the data from the source domain and only 4 samples per class from the target domain $\{S(x_t, y_t)_{t=1}^{N_S}, T(x_t, y_t)_{t=1}^{4n}\}$ where n represents the number of unique labels in the target domain distribution. It has been shown that

human categorization often asymptotes after just three or four examples Xu and Tenenbaum [2007]. Our goal is to build a classifier with almost human level capability to predict the remaining labels $T(y_t)_{t=1}^{N_T-4n}$ in the target domain given $T(x_t)_{t=1}^{N_T-4n}$.

3.2 Training objectives

There are three classifiers in the proposed model each with its training objective (See supplementary material for more details). The overall training objective however is given by:

$$\begin{aligned} \underset{\theta_1,\theta_2,\theta_3}{argmin}\, E(\theta_1,\theta_2,\theta_3) = \frac{1}{N+4n} \sum_{i=1....N+4n} L(y_i, f(X_i;\theta_3)| \\ \frac{1}{4n} \sum_{i=1....4n} L(y_i, f(X_i;\theta_2)| \\ \frac{1}{N} \sum_{i=1....N} L(y_i, f(X_i;\theta_1)))) \end{aligned} \tag{1}$$

4 Dataset

We gathered dataset of beef meats classified broadly into four groups of excellent, good, acceptable and spoiled with all the datasets skewed towards the spoiled meat. These data have been collected with the aid of electronic nose gas sensors and other sensors to measure variables such as humidity, temperature and TVC (continuous label of microbial population). Each data point in the datasets was recorded per minute in a sequential manner. **Dataset 1** is made up of time series data of beef quality collected across five different instances across two years. Each data instance is 2160 in length Wijaya et al. [2018b]. **Dataset 2** consist of an extra-lean fresh beef monitored for about 75 minutes under fluctuating conditions of humidity and temperature Wijaya et al. [2018a, 2017a, 2016, 2017b]. **Dataset 3** contains 12 files of different beef meat cuts such as Inside - Outside, Round, Top Sirloin among others. Eleven gas sensors were used to collect the data Wijaya [2018].

5 Model, Procedures, Experiments & Baselines

We consider experiments (154 in all) across all datasets where we aim to investigate the performance of our model across a variety of contexts. We aim to evaluate the performance of our model when there is significant difference in the distribution of the input features and labels across the source and target domains.

5.1 Model architecture

We use four recurrent neural networks, LSTM Hochreiter and Schmidhuber [1997] overall (two for training the two clusters of the source data alone and another two for training after adding the few target data) with four cells each and one logistic regression classifier. We use logistic regression to train the few labeled target data with the cluster labels as the input size here is too small for a neural network. The LSTMs with four cells each all employ a many to one classifier with a time-step of two. We Implement the model using Keras and Scikit learn.

5.2 Training procedure

To train the classifier, we use the cluster label where the probability of predicting correctly the label of the data is maximized (two clusters constructed from the source data with the aid of Gaussian mixture model). In situations where none of the clusters can predict correctly any of the training target data label, we use the cluster label where the frequency of the label is higher. We run the model ten times settling for the iteration that performed well on the few target data in their new local domain. This model is run a further five times on the unseen target data to find the average classification accuracy.

5.3 Baselines

We use logistic regression (LR), Ada-boost (AB) with a hundred estimators, Support vector machine (SVM), Semi Supervised learning (SS) Wei and Keogh [2006], Deep Neural Network (DNN) with two layers each with 256 neurons, Long short term memory (LSTM) with one layer and 4 cells, Recurrent Domain Adaptation Neural Network (RDANN) Ganin et al. [2016].

6 Results

Results (Table 1), show that our approach outperforms all baselines most of the times and overall across all experiments. Due to lack of the right quantity of data, most of the deep learning models appear to perform poorly.

Source-Target	LR	AB	SVM	SS	DNN	LSTM	R-DANN	Ours
$1_{1-5} - 2$	19.59	69.43	11.26	36.24	5.95	46.02	47.18	**79.85**
$1_1 - 3_{1-12}$	46.58	22.00	33.60	14.65	5.30	37.55	44.86	**65.07**
$1_2 - 3_{1-12}$	33.97	54.73	25.89	13.59	7.97	41.21	37.61	**64.39**
$1_3 - 3_{1-12}$	36.65	54.73	26.37	13.34	3.54	60.62	33.79	**57.73**
$1_4 - 3_{1-12}$	39.41	31.37	28.42	13.63	10.53	13.31	32.62	**66.77**
$1_5 - 3_{1-12}$	58.99	64.53	30.02	69.32	12.24	23.74	35.43	**69.90**
$2 - 1_{1-5}$	52.06	59.44	49.03	38.81	11.69	14.08	42.77	**67.14**
$2 - 3_{1-12}$	75.41	**83.82**	73.51	27.11	7.21	61.60	63.51	78.59
$3_1 - 1_{1-5}$	54.14	46.66	45.31	**64.46**	8.34	49.06	38.91	52.19
$3_2 - 1_{1-5}$	59.34	57.83	45.09	**59.45**	19.44	44.27	38.91	54.85
$3_3 - 1_{1-5}$	58.08	62.22	44.99	**71.04**	19.44	51.30	42.26	52.84
$3_4 - 1_{1-5}$	**63.36**	62.22	47.75	47.66	14.27	43.67	39.37	62.98
$3_5 - 1_{1-5}$	61.99	**62.22**	43.88	41.04	15.58	50.44	34.33	59.89
$3_6 - 1_{1-5}$	54.97	**62.22**	44.56	34.50	14.38	45.37	37.59	60.23
$3_7 - 1_{1-5}$	55.84	62.22	48.28	20.72	14.26	39.43	41.79	**65.33**
$3_8 - 1_{1-5}$	48.93	62.22	47.35	59.46	22.07	37.93	45.19	**62.39**
$3_9 - 1_{1-5}$	53.40	62.22	43.48	18.96	20.17	33.31	47.12	**64.46**
$3_{10} - 1_{1-5}$	58.65	62.22	47.80	**68.15**	15.52	47.32	30.66	61.37
$3_{11} - 1_{1-5}$	55.69	62.22	43.67	**74.14**	13.31	48.44	35.92	56.77
$3_{12} - 1_{1-5}$	74.21	51.11	44.01	**100**	14.69	34.20	36.69	63.46
$3_{1-12} - 2$	51.56	88.08	60.28	91.33	20.52	33.60	78.74	**92.18**
Avg	52.99	59.23	42.12	46.55	13.16	40.78	43.59	**66.59**

Table 1: Classification accuracy all in %. Our approach can be seen to outperform all baselines most of the time across all experiments suggesting that it is useful when there are both label shifts as well as covariate shifts in the input features.

7 Conclusion

In this paper we have introduced a new approach towards transferring knowledge from one time-series domain to another using only few samples for the purpose of assessing beef quality. Our approach leverages the construction of unsupervised classification tasks to improve actual beef quality classification tasks. We evaluate our approach on a time series data of beef quality cuts collected across different conditions. Results across a variety of experiments show that our approach performed competitively compared to competing baselines most especially when the distribution of the target domain labels differs significantly from that of the source domain. Our work is without its limitations, due to the number of experiments carried out and the total number of neural networks deployed, we have used the same hyper-parameters across all experiments. Careful tuning of the networks or change of architecture in the future can generate better results. In addition, Just like any other hierarchical model, this approach incurs additional computational cost. Furthermore, we envisage distributions with more classes can benefit from deep hierarchical clustering Heller and Ghahramani [2005] compared to the flat clustering we have used. Future work could investigate a combination of some of the techniques used here together with adversarial domain adaptation methods.

References

G. A. United Nations. Transforming our world: the 2030 agenda for sustainable development. New York: United Nations, 2015.

María L Rodríguez-Méndez, José A De Saja, Rocio González-Antón, Celia García-Hernández, Cristina Medina-Plaza, Cristina García-Cabezón, and Fernando Martín-Pedrosa. Electronic noses and tongues in wine industry. *Frontiers in bioengineering and biotechnology*, 4:81, 2016.

Rahman Wijaya, Riyanarto Sarno, and Enny Zulaika. Electronic nose dataset for beef quality monitoring under an uncontrolled environment?, mendeley data, v2, 2018a.

Sanjay Purushotham, Wilka Carvalho, Tanachat Nilanon, and Yan Liu. Variational recurrent adversarial deep domain adaptation. *International Conference on Learning Representations*, 2016.

Qingchang Zhu, Zhenghua Chen, and Chai Soh Yeng. A novel semi-supervised deep learning method for human activity recognition. *IEEE Transactions on Industrial Informatics*, 2018.

Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. *arXiv preprint arXiv:1904.06487*, 2019.

Fei Xu and Joshua B Tenenbaum. Word learning as bayesian inference. *Psychological review*, 114 (2):245, 2007.

Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. The ucr time series classification archive. 2015.

Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Transfer learning for time series classification. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 1367–1376. IEEE, 2018.

Li Wei and Eamonn Keogh. Semi-supervised time series classification. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 748–753. ACM, 2006.

Donghai Guan, Weiwei Yuan, Young-Koo Lee, Andrey Gavrilov, and Sungyoung Lee. Activity recognition based on semi-supervised learning. In *13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 2007)*, pages 469–475. IEEE, 2007.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

Rahman Wijaya, Riyanarto Sarno, and Enny Zulaika. Electronic nose dataset for beef quality monitoring under an uncontrolled environment?, mendeley data, v3. *Mendeley Data*, 2018b.

Dedy Rahman Wijaya, Riyanarto Sarno, and Enny Zulaika. Information quality ratio as a novel metric for mother wavelet selection. *Chemometrics and Intelligent Laboratory Systems*, 160:59–71, 2017a.

Dedy Rahman Wijaya, Riyanarto Sarno, and Enny Zulaika. Sensor array optimization for mobile electronic nose: Wavelet transform and filter based feature selection approach. *International Review on Computers and Software*, 11(8):659–671, 2016.

Dedy Rahman Wijaya, Riyanarto Sarno, Enny Zulaika, and Shoffi Izza Sabila. Development of mobile electronic nose for beef quality monitoring. *Procedia Computer Science*, 124:728–735, 2017b.

Dedy Rahman Wijaya. Dataset for electronic nose from various beef cuts, 2018. URL `https://doi.org/10.7910/DVN/XNFVTS`.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Katherine A Heller and Zoubin Ghahramani. Bayesian hierarchical clustering. In *Proceedings of the 22nd international conference on Machine learning*, pages 297–304. ACM, 2005.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

Supplementary Material

a Approach: Additional Information

The proposed approach leverages the construction of auxiliary tasks to improve the performance of a downstream supervised learning task. The essence of constructing auxiliary tasks is to aid the efficiency of learning a similar or related task. To construct auxiliary tasks for the purpose of our approach, we aim to find clusters in the source domain data where the probability of classifying each of the few labeled target data is maximized. The task therefore is defined as given the cluster label C_k where the probability of classifying the few target labels is maximized, find $\underset{\hat{y}}{argmax}\ p(\hat{y}|\hat{X}, C_k)$. The choice of the number of clusters is an open question. But it is essential to find a balance between the difficulty of finding $p(C_k|\hat{X}_{i=1...N_t})$ and that of $\underset{\hat{y}}{argmax}\ p(\hat{y}|\hat{X}_{i=1...N_t}, C_k)$.

There are four benefits of our approach with respect to domain adaptation. First, by finding clusters in the source distribution features, we are able to reduce the mismatch in the distribution of labels. Second, by allocating the target data to source models or cluster where its probability of being predicted is maximised, we reduce the mismatch in the feature distribution between the source and target domain. Third, since sensor data are extremely noisy, our approach has the potential to ensure extremely noisy inputs are represented in clusters where they appear as outliers enabling the efficient learning of the model parameters. And lastly, by only using labels for the classes that are far apart in the feature space, it is not necessary in some cases to obtain sample labels of the target data for all the classes as similar input features will be found in the same cluster attached to the same model.

b Algorithm

Algorithm 1

1: **Input:** Source data: $S(x_t, y_t)_{t=1}^{N_S}$, Target data: $T(\hat{x}_t, \hat{y}_t)_{t=1}^{4n}$
2: **Output:** Target domain class labels, $\hat{y}^1, \hat{y}^2, ..., \hat{y}^N$
3: Find clusters $C_{k=1....k_n}$ in the input dataset.
4: Train each cluster C_k with a RNN model M_i.
5: Find the cluster (C_k) / model (M_k) where $\underset{\hat{y}}{argmax}\ p(\hat{y}|\hat{X}, C_k)$
6: Retrain each of the RNN model M_k again with the old source data in C_k combined with the new data from $T(\hat{x}_t, \hat{y}_t)_{t=1}^{4n}$
7: Train a classifier to assign target data into the right cluster / model using $T(\hat{x}_t)_{t=1}^{4n}$ and labels from (C_k) .
8: **for** each datapoint in test data **do**:
9: Assign data to model M_k from step **6** using classifier from step **7**.
10: Run the RNN model M_k attached to the assigned cluster C_k from step **6**.
11: **end**
12: **return** $\hat{y}_1, \hat{y}_2, ..., \hat{y}_N$

c Training objectives: Additional Information

There are three classifiers in the proposed model each with its training objective. The training objective for classifying source domain labels alone is given by:

$$\underset{\theta_1}{argmin}\ E(\theta_1) = \frac{1}{N} \sum_{i=1....N} L(y_i, f(X_i; \theta_1)) \tag{2}$$

The training objective for the local domain classification (training few labeled target data with cluster labels) is given by :

$$\underset{\theta_2}{argmin}\ E(\theta_2) = \frac{1}{4n} \sum_{i=1....4n} L(y_i, f(X_i; \theta_2)) \tag{3}$$

Where n is the number of unique classes in the target domain. The loss for classifying labels from all the local domains after adding few labeled data from the target domain (both source and target labels) is given by:

$$\underset{\theta_3}{argmin}\ E(\theta_3) = \frac{1}{N+4n} \sum_{i=1....N+4n} L(y_i, f(X_i; \theta_3)) \tag{4}$$

The overall training objective is given by training objective (equation 4) conditioned on training objective (equation 3) which is conditioned on training objective (equation 2).

$$\begin{aligned} \underset{\theta_1,\theta_2,\theta_3}{argmin}\ E(\theta_1, \theta_2, \theta_3) = \frac{1}{N+4n} \sum_{i=1....N+4n} L(y_i, f(X_i; \theta_3)| \\ \frac{1}{4n} \sum_{i=1....4n} L(y_i, f(X_i; \theta_2)| \\ \frac{1}{N} \sum_{i=1....N} L(y_i, f(X_i; \theta_1)))) \end{aligned} \tag{5}$$

d Model Architecture

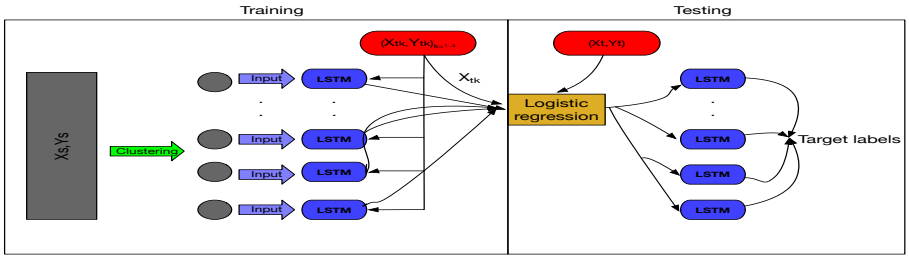

Figure 1: Model architecture showing the structure of our approach.

e Baselines: Additional Information

We compare our approach with several baselines with and without domain adaptation described below. While the domain adaptation baselines are fully unsupervised with the advantage of access to all target features during training, we still aim to compare these methods with our approach to see how these constraints influence performance.

Logistic regression (LR): We use a multinomial variant with a lbfgs solver.

Adaboost (AB): With 100 estimators.

Support Vector Machine (SVM): One versus one. We add the few labeled data from the target distribution to the training data here and also for LR and AB above.

Semi-Supervised (SS): Uses one nearest neighbour Wei and Keogh [2006] classifier. We use a variant of this approach where we extend the original approach to a four way classifier since the original approach was proposed for binary classification. We build a dedicated classifier (assumed to be perfect) for each of the classes containing data from the source domain. We classify each data in the test data by assigning data into class where the one nearest neighbour has the minimum distance with respect

to the four classifiers. Test data is added to the training data if the distance to the nearest neighbour is smaller than the mininum distance between samples in the same class in the training dataset. We use only the training data here to assess the ability of this approach to generalise when used on datasets from another domain while the test data is added to the training data as discussed above during testing.

Deep Neural Network (DNN) : With two layers and 256 neurons each, trained over 100 epochs with dropout = 0.2, softmax layer and adam opimizer Kingma and Ba [2015].

Recurrent Neural Network (LSTM): A long short term memory (LSTM) network with 1 layer, timestep = 2, four cells, trained over 100 epochs with dropout = 0.2, softmax layer and adam optimizer Kingma and Ba [2015]. We use both training data and few labeled target data for training here and also for DNN.

Recurrent Domain Adaptation Neural Network (R-DANN): This is the domain adaptation approach of Ganin et al. [2016] but with an LSTM in the feature extractor as in Purushotham et al. [2016]. Two layer feed-forward network with 128 neurons each are further added to the feature extractor as well as the source and domain classifiers. Relu activation is used throughout the feature extraction network and tanh for the LSTM with a softmax layer for classification.

f Data & Preprocessing: Additional Information

We provide more information on the datasets we have used here.

We gathered dataset of beef meats classified broadly into four groups of excellent, good, acceptable and spoiled with all the datasets skewed towards the spoiled meat. These data have been collected with the aid of electronic nose gas sensors and other sensors to measure variables such as humidity, temperature and TVC (continuous label of microbial population). Each data point in the datasets was recorded per minute in a sequential manner.

Dataset 1: This consists of time series data of beef quality collected across five different instances across two years. Each data instance is 2160 in length. Nine gas sensors (MQ135, MQ136, MQ2, MQ3, MQ4, MQ5, MQ6, MQ8, MQ9) were used to collect the data including the humidity and temperature sensors Wijaya et al. [2018b].

Dataset 2: This contains an extra-lean fresh beef monitored for about 75 minutes under fluctuating conditions of humidity and temperature. Ten Gas Sensors (MQ135, MQ136, MQ2, MQ3, MQ4, MQ5, MQ6, MQ7, MQ8, MQ9) were used to collect the data as well as humidity and temperature sensors Wijaya et al. [2018a, 2017a, 2016, 2017b].

Dataset 3: Contains 12 files of different beef meat cuts such as Inside - Outside, Round, Top Sirloin among others. Eleven gas sensors (MQ135, MQ136, MQ137, MQ138, MQ2, MQ3, MQ4, MQ5, MQ6, MQ8, MQ9) were used to collect the data Wijaya [2018].

To ensure the input features are uniform across all datasets collected, we remove the features corresponding to the humidity and temperature variable as well as those corresponding to the sensors MQ7, MQ138 and MQ137.

g Evaluation: Additional Information

All results are reported using the source data and the few target training data except for SS and R-DANN to demonstrate their inherent limitations in the absence of target domain data. We evaluate all methods on the average classification accuracy from one dataset to another. For example, $2 - 1_{1-5}$ means model trained on dataset 2 containing just one file is tested on dataset 1 with five datasets or files. The average classification accuracy is thus based on the average of the accuracies of the model trained on dataset 2 and tested on the five datasets in dataset 1.

Dataset	Data length	Beef cut	Distribution of classes			
			Excellent (Class 1)	Good (Class 2)	Acceptable (Class 3)	Spoiled (Class 4)
Dataset1	2160	-	0.111	0.306	0.139	0.444
	2160	-	0.167	0.250	0.277	0.306
	2160	-	0.168	0.250	0.167	0.417
	2160	-	0.168	0.250	0.194	0.389
	2160	-	0.168	0.250	0.194	0.389
Dataset2	4453	Extra-lean	0.063	0.046	0.040	0.851
Dataset3	2220	Inside Outside	0.0008	0.0005	0.0005	0.998
	2220	Round	0.0008	0.0005	0.0005	0.998
	2220	Top Sirloin	0.135	0.162	0.108	0.595
	2220	Tenderloin	0.135	0.162	0.108	0.595
	2220	Flap meat	0.135	0.162	0.108	0.595
	2220	Striploin	0.135	0.162	0.108	0.595
	2220	Rib eye	0.135	0.162	0.108	0.595
	2220	Skirt meat	0.135	0.162	0.108	0.595
	2220	Brisket	0.135	0.162	0.108	0.595
	2220	Clod Chuck	0.135	0.162	0.108	0.595
	2220	Shin	0.135	0.162	0.108	0.595
	2220	Fat	0.108	0.135	0.162	0.595

Table 2: Distribution of all datasets showing the length, beef cut and the distribution of the different classes of beef quality across three datasets. It can be seen that the distribution of the classes is skewed towards the spoiled meat.