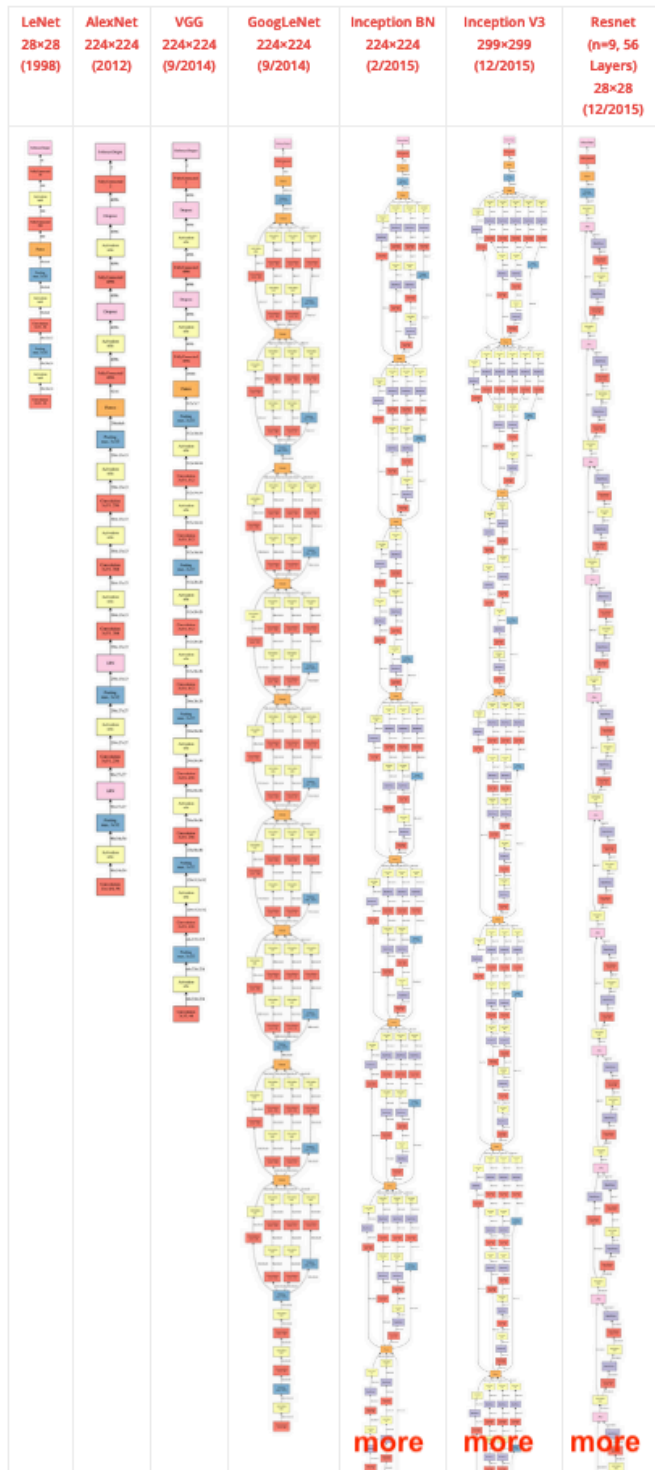
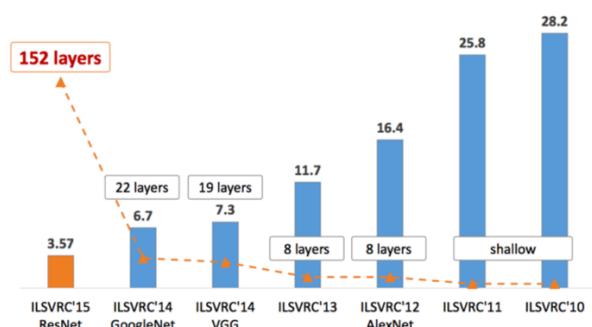


1. 請說明你實作的 CNN 模型(best model)，其模型架構、訓練參數量和準確率為何？(1%)

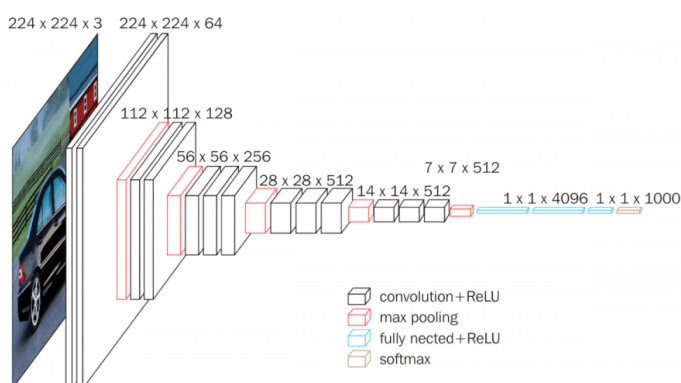
本次實作的模型參考自經典的影像辨識 model vgg16 [1]，相較其他許多更深的網路架構，如下圖 [2]，vgg 是相對好訓練的模型，深度淺參數較少，架構簡單容易自己接，我的 GPU 的記憶體(Nvidia 1660 6 GiB)也比較塞得下



且 vgg 在 2014 年的 ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 中的表現也相當不錯，是 error rate 有所突破的一個轉捩點，如下圖[3]

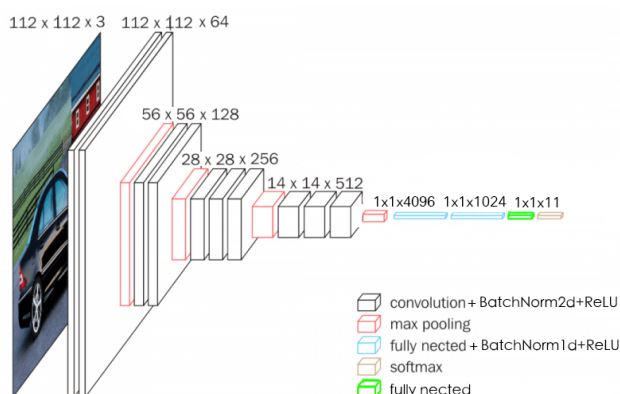


下圖為原始的 vgg16，input 為 224x224pixels x 3channels 的圖片，共經過 13 層的 convolutional layer，和 3 層的 fully connected layer，output 為 1000 個分類。

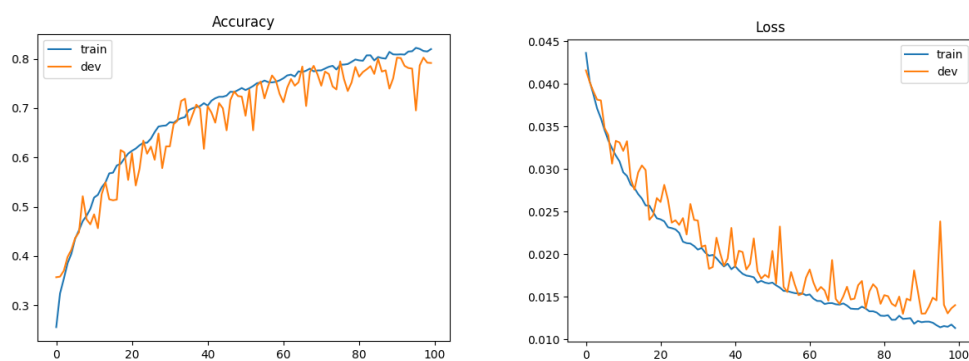


而我們的 dataset 多為 512x512pixels x 3channels 的圖片，需經過 resize 至 224x224pixels x 3channels，並修改最後一層 linear network 的輸出為 11 維，然而因為參數眾多，batch size 只能開到 8，train 一個 epoch 的時間要 340s 左右，曠日費時，每調整一個參數或換一個 optimizer 都要花大量的時間等待。

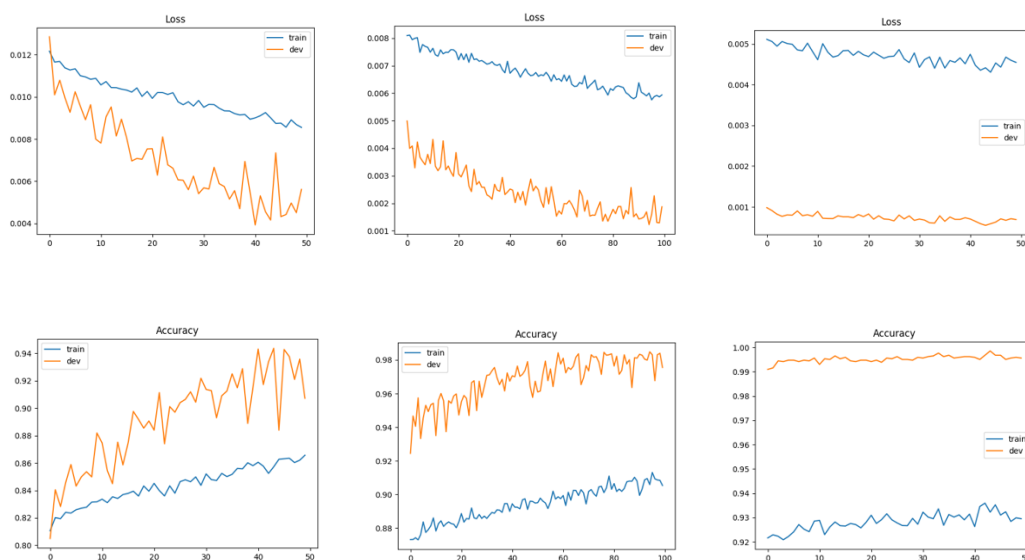
因此本次作業中將 vgg16 輕量化，把輸入的圖片 resize 至 112x112pixels x 3channels，並減少三層中間的 convolutional layer，也減少後面 fully connected layer 的 neuron 數量，batch size 可開到 48，一個 epoch 只花 70s 左右



並有在最後兩個 FC 做 dropout 50%，來改善 validation set 的 accuracy，並使用 Stochastic gradient descent (SGD) with learning rate = 0.002 and momentum = 0.9，training 100 個 epoch，在 training&validation set 上的 accuracy>0.8 都有 beat strong baseline，詳細的結果如下圖：



得到此 model 後，基於這個 weight 我們再加入 validation set 的資料一起進行訓練 200 個 epoch，由左到右共分為 0~50 with lr = 0.001，50~150 with lr=0.0005，150~200 with lr=0.0001，詳細結果如下圖(dev cure 可忽略)：

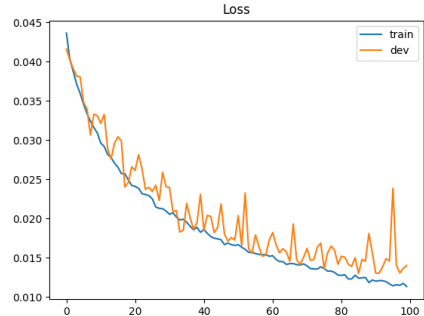
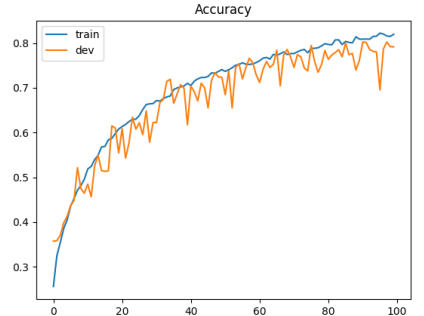
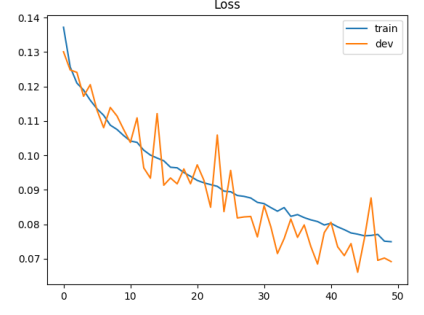
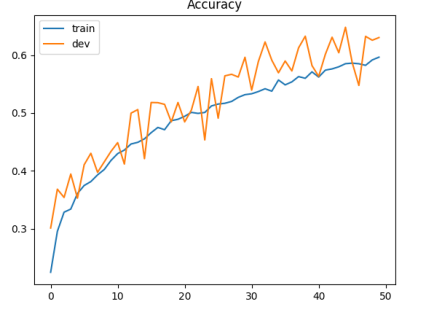


而在 kaggle public leaderboard 上的 score 為

Model and Dataset	Number of epoch	Validation score	Kaggle public score
vgg16lite on train	100	0.823032	0.85056 (V)
vgg16lite on train+val	50	-	0.87985
vgg16lite on train+val	100	-	0.88164
vgg16lite on train+val	150	-	0.89719 (V)
vgg16lite on train+val	200	-	0.89420

2. 請實作與第一題接近的參數量，但 CNN 深度（CNN 層數）減半的模型，並說明其模型架構、訓練參數量和準確率為何？(1%)

Model	Number of parameters	Number of layers	Structure (conv kernel = 3x3)	Batch size / Time per ep	Accuracy on valid
vgg16lite	114622027	13	conv64, conv64, conv128, conv128, conv256, conv256, conv256, conv512, conv512, conv512, 3*fc	48, 70s	0.723324
vgg16lite-shallow	114659083	8	conv128, conv512, conv512, conv512, conv512, 3*fc	16, 170s	0.630321

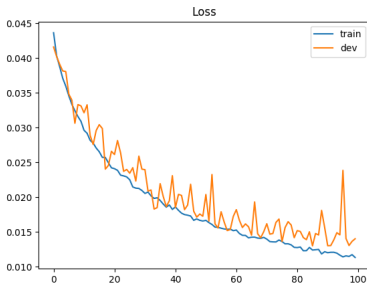
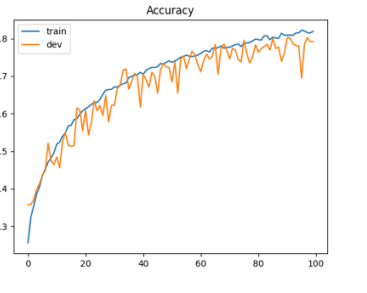
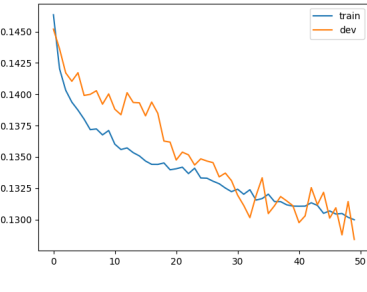
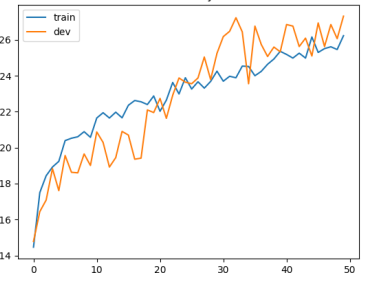
Model	Loss curve	Accuracy curve
vgg16lite		
vgg16lite-shallow		

根據第一題的模型，把 convolution 的層數減半，調整每層的 dimension 使保持總參數差不多一樣，其他條件不變，然後 train on training set 共 50 個 epoch。根據實驗結果觀察，vgg16lite-shallow 在參數量幾乎和 vgg16lite 一樣甚至較多一點點的情況下，performance 仍沒有網路架構比較深的 vgg16lite 來得好，這是因

為每一層的 convolution 都是在提取前一層的 feature，較深的網路會基於前面提取的 feature 繼續提取那些 feature 的 feature，進而一直傳承下去，也就是 modularization 的概念，而層數較淺網路模組化的能力就較弱，因此分類的效果就沒這麼好，相關 Fat + Short v.s. Thin + Tall 的討論可以參考論文[4]。

3. 請實作與第一題接近的參數量，簡單的 DNN 模型，同時也說明其模型架構、訓練參數和準確率為何？(1%)

Model	Number of parameters	Number of layers	Structure (conv kernel = 3x3)	Batch size / Time per ep	Accuracy on valid
vgg16lite	114622027	13	conv64, conv64, conv128, conv128, conv256, conv256, conv256, conv512, conv512, conv512, 3*fc	48, 70s	0.723324
dnn	148244875	12	fc64*7*7, fc64*7*7, fc4096, fc1024, fc1024, fc512, fc512, fc512, fc256, fc128, fc11	48, 50s	0.273178

Model	Loss curve	Accuracy curve
vgg16lite		
dnn		

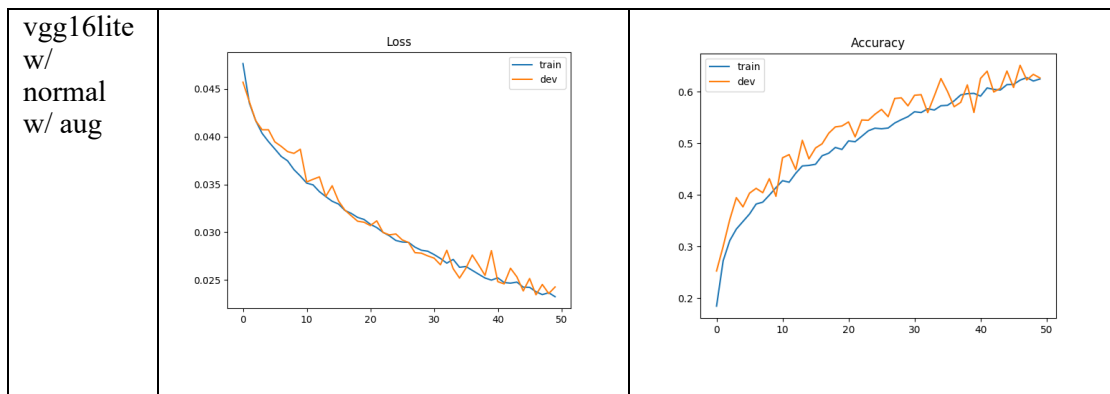
根據第一題的模型，把 convolution 的層和 maxpooling 全部拿掉加入更多的 fully connected layer，調整每層的 dimension 使保持總參數差不多一樣，其他條件不變，然後 train on training set 共 50 個 epoch。根據實驗結果觀察，dnn 在參數量幾乎和 vgg16lite 一樣甚至較多一點點的情況下，performance 仍沒有使用 convolutional layer 的 vgg16lite 來得好，這是因為影像辨識的問題符合三個 property：1. Some patterns are much smaller than the whole image，所以用 Conv 會比用 FC 來得有效率不用一次把整張圖片一起看。2. The same patterns appear in different regions，所以用 FC 訓練會不容易找到合適的 feature (weight)。3. Subsampling the pixels will not change the object，所以適合用 maxpooling 來增大視野在不同 scale 下找 feature。

4. 請說明由 1~3 題的實驗中你觀察到了什麼？(1%)

乘上兩題，根據上述原因，在做 deep learning 時還是設計 Thin + Tall 的 network 比 Fat + Short 的 network 來得好。另外在進行 image recognition 或 speech recognition 這種滿足三個 property：1. Some patterns are much smaller than the whole image, 2. The same patterns appear in different regions, 3. Subsampling the pixels will not change the object 的情況時，適合用 maxpooling 和 convolutional layer 構成的 CNN 來達到比只有 fully connected layer 的 DNN 更好的效果。

5. 請嘗試 data normalization 及 data augmentation，說明實作方法並且說明實行前後對準確率有什麼樣的影響？(1%)

Model	Loss curve	Accuracy curve
vgg16lite w/o normal w/ aug		
vgg16lite w/o normal w/o aug		



data augmentation:

加入 data augmentation(torchvision.transform 的 RandomRotation, RandomCrop, RandomFilp)後在 validation set 上的 accuracy 會變好，因為能避免 model 錯誤學習到 feature 的角度或是位置而不是 feature 本身，如果沒有做 data augmentation 的話會很明顯的 overfit training set，導致 validation 的 loss 降不下去，accuracy 也上不去，因為在下一次沒看過的圖片進來時，有同樣的 feature 但不同的角度時的分類能力很弱。

data normalization:

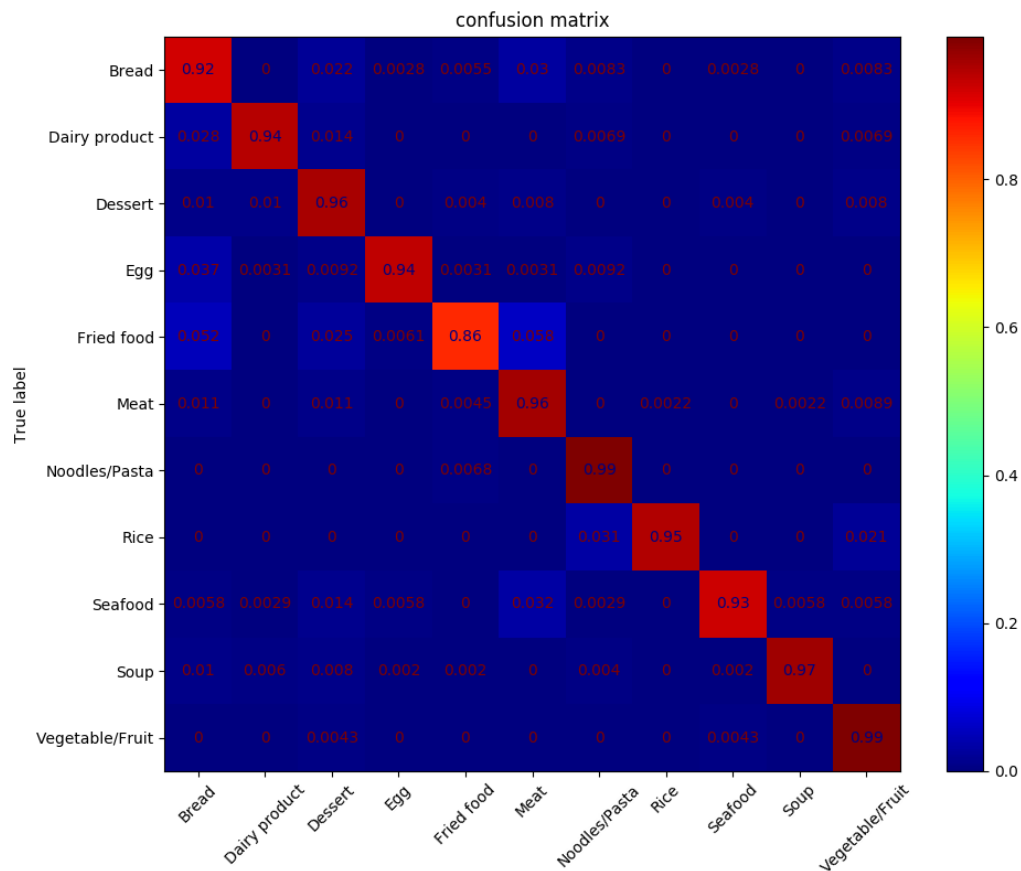
把 data toTensor 後，會 converts a PIL Image or numpy.ndarray (H x W x C) in the range [0, 255] to a torch.FloatTensor of shape (C x H x W) in the range [0.0, 1.0]，針對這個被 scaling 到 0~1 區間的值，我們可以算每個 channel 的 mean 和 standard deviation 用手上有的所有資料(test, train, valid)，結果為：

Mean: tensor([0.3339, 0.4526, 0.5676])

Std: tensor([0.2298, 0.2322, 0.2206])

然後再用 torchvision.transform 的 normalize 去做 normalization，因為把 data 標準化後，不同的 channel(feature) 的差異就有相同的 scaling，差異量很大的跟差異量微小的 feature 會有一樣的影響力，理論上在各個 feature 數值範圍差異很大時會很有用，更易於 gradient decent 的收斂，找到 argmin。但是實際的結果中並沒有比較好，原因可能在於影像的數值皆已被縮放到 0~1 有相同的變動區間，再去做 normalization 會抹殺掉不同 feature 間的差異量差距，可能會因此對原本應該被放大檢視或縮小檢視的 feature 一視同仁，所以 accuracy 沒有比較好。

6. 觀察答錯的圖片中，哪些 class 彼此間容易用混？[繪出 confusion matrix 分析](1%)



上圖是由 validation set 上的結果畫出來的，看起來是分類都分得不錯，唯一低於 0.9 的是炸物(fried food)的分類，稍微容易和麵包(Bread)跟肉(Meat)分類，就照片來看也算是合理，因為這三者的顏色其實還蠻相近的，但整體來說還是分得很開拉。

[1] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).

[2] <https://josephpcohen.com/w/visualizing-cnn-architectures-side-by-side-with-mxnet/>

[3] <https://medium.com/analytics-vidhya/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5>

[4] Seide, Frank, Gang Li, and Dong Yu. "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks." *Interspeech*. 2011.