An i-Vector based Approach to Training Data Clustering for Improved Speech Recognition

Yu Zhang[1,2]*, Jian Xu*[1,3]*, Zhi-Jie Yan*[1]*, Qiang Huo*[1]

[1]Microsoft Research Asia, Beijing, China
[2]Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
[3] Department of Automation, University of Science and Technology of China, Hefei, China
sjtuzy@gmail.com, {v-jiaxu, zhijiey, qianghuo}@microsoft.com

Abstract

We present a new approach to clustering training data for improved speech recognition. Given a training corpus, a so-called i-vector is extracted from each training utterance. A hierarchical divisive clustering algorithm is then used to cluster the training i-vectors into multiple clusters. For each cluster, an acoustic model (AM) is trained accordingly. Such trained multiple AMs can then be used in recognition stage to improve recognition accuracy. The proposed approach is very efficient therefore can deal with very large scale training corpus on current mainstream computing platforms. We report experimental results on a voice search task with 7,500 hours of speech training data.
Index Terms: factor analysis, data clustering, acoustic model

1. Introduction

Training multiple sets of acoustic model (AM) to improve recognition accuracy by clustering training data was an old research topic in the field of automatic speech recognition (ASR). Recently it attracts renewed interest because increasingly more training data collected from a very large population in diversified acoustic environments and transmission channels is becoming available to build ASR systems. An interesting study on this topic was reported in [1] for a voice search task. In this paper, we present a similar study with a new data clustering approach based on a so-called i-vector technique [2]. The proposed approach is very efficient therefore can deal with very large scale training corpus on current mainstream computing platforms.

The rest of the paper is organized as follows. In Section 2, we present i-vector extraction approach. In Section 3, we describe our i-vector based data clustering approach for improved ASR. In Section 4, we report preliminary experimental results on a voice search task with 7,500 hours of speech training data. Finally, we conclude the paper in Section 5.

2. i-Vector Approach

In [2], an i-vector extraction approach was described, but important information on how to estimate hyperparameters (a.k.a. total variability matrix) was missing. Readers were referred to [3] for such technical details instead. However, because so-called "Baum-Welch" statistics (instead of "Viterbi" ones) were used to extract an i-vector from each utterance, the theoretical justification and derivation in [3] cannot be used to justify the practice in [2] any more. In the following subsections, we explain the theoretical justification of the i-vector extraction approach borrowed from [2] and present our version of hyperparameter estimation procedure.

2.1. Data Model

Suppose we are given a set of training data denoted as $\mathcal{Y} = \{\boldsymbol{Y}_i | i = 1, 2, \ldots, I\}$, where $\boldsymbol{Y}_i = (\boldsymbol{y}_1^{(i)}, \boldsymbol{y}_2^{(i)}, \ldots, \boldsymbol{y}_{T_i}^{(i)})$ is a sequence of D-dimensional feature vectors extracted from the i-th training utterance. From $\mathcal{Y}$, a Gaussian mixture model (GMM) can be trained using a maximum likelihood (ML) approach to serve as a so-called universal background model (UBM):

$$p(\boldsymbol{y}) = \sum_{k=1}^{K} c_k \mathcal{N}(\boldsymbol{y}; \boldsymbol{m}_k, \boldsymbol{R}_k) \tag{1}$$

where c_k's are mixture coefficients, $\mathcal{N}(\cdot; \boldsymbol{m}_k, \boldsymbol{R}_k)$ is a normal distribution with a D-dimensional mean vector $\boldsymbol{m}_k$ and a $D \times D$ diagonal covariance matrix $\boldsymbol{R}_k$. Let $\boldsymbol{M}_0$ denote the $(D \cdot K)$-dimensional supervector by concatenating the $\boldsymbol{m}_k$'s and $\boldsymbol{R}_0$ denote the $(D \cdot K) \times (D \cdot K)$ block-diagonal matrix with $\boldsymbol{R}_k$ as its k-th block component. Let's use $\Omega = \{c_k, \boldsymbol{m}_k, \boldsymbol{R}_k | k = 1, \ldots, K\}$ to denote the set of UBM-GMM parameters.

2.2. i-Vector Extraction

Given an utterance $\boldsymbol{Y}_i$, let's use a $(D \cdot K)$-dimensional random supervector $\boldsymbol{M}(i)$ to characterize its variability independent of linguistic content, which relates to $\boldsymbol{M}_0$ as follows:

$$\boldsymbol{M}(i) = \boldsymbol{M}_0 + \boldsymbol{T}\boldsymbol{w}(i) \tag{2}$$

where $\boldsymbol{T}$ is a fixed but unknown $(D \cdot K) \times F$ rectangular matrix of low rank (i.e., $F \ll (D \cdot K)$), and $\boldsymbol{w}(i)$ is an F-dimensional random vector having a prior distribution of standard normal distribution $\mathcal{N}(\cdot; \boldsymbol{0}, \boldsymbol{I})$. A graphical model representation is shown in Fig. 1. In [2], $\boldsymbol{T}$ is called the total variability matrix. Given $\boldsymbol{Y}_i$, Ω, and $\boldsymbol{T}$, the so-called i-vector defined in [2] is actually the solution of the following problem:

$$\hat{\boldsymbol{w}}(i) = \underset{\boldsymbol{w}(i)}{\operatorname{argmax}} \prod_{t=1}^{T_i} \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{y}_t^{(i)}; \boldsymbol{M}_k(i), \boldsymbol{R}_k)^{P(k|\boldsymbol{y}_t^{(i)}, \Omega)} p(\boldsymbol{w}(i)) \tag{3}$$

where $\boldsymbol{M}_k(i)$ is the k-th D-dimensional subvector of $\boldsymbol{M}(i)$, and

$$P(k|\boldsymbol{y}_t^{(i)}, \Omega) = \frac{c_k \mathcal{N}(\boldsymbol{y}_t^{(i)}; \boldsymbol{m}_k, \boldsymbol{R}_k)}{\sum_{l=1}^{K} c_l \mathcal{N}(\boldsymbol{y}_t^{(i)}; \boldsymbol{m}_l, \boldsymbol{R}_l)}.$$

The closed-form solution of the above problem gives the i-vector extraction formula as follows:

$$\hat{\boldsymbol{w}}(i) = \boldsymbol{l}^{-1}(i) \boldsymbol{T}^{\top} \boldsymbol{R}_0{}^{-1} \boldsymbol{\Gamma}_{\boldsymbol{y}}(i) \tag{4}$$

This work was done when Yu Zhang and Jian Xu were interns in Microsoft Research Asia, Beijing, China.

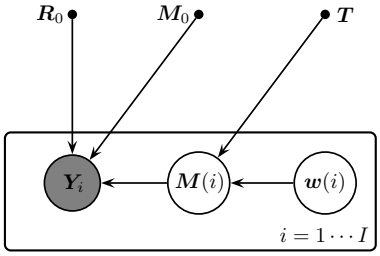

Figure 1: *A graphical model representation of i-vector approach.*

where

$$\boldsymbol{l}(i) = \boldsymbol{I} + \boldsymbol{T}^\top \boldsymbol{\Gamma}(i) \boldsymbol{R}_0^{-1} \boldsymbol{T}; \tag{5}$$

$\boldsymbol{\Gamma}(i)$ is a $(D \cdot K) \times (D \cdot K)$ block-diagonal matrix with $\gamma_k(i)\boldsymbol{I}_{D\times D}$ as its k-th block component; $\boldsymbol{\Gamma}_{\boldsymbol{y}}(i)$ is a $(D \cdot K)$-dimensional supervector with $\Gamma_{\boldsymbol{y},k}(i)$ as its k-th D-dimensional subvector. The "Baum-Welch" statistics $\gamma_k(i)$ and $\Gamma_{\boldsymbol{y},k}(i)$ are calculated as follows:

$$\gamma_k(i) = \sum_{t=1}^{T_i} P(k|\boldsymbol{y}_t^{(i)}, \Omega) \tag{6}$$

$$\Gamma_{\boldsymbol{y},k}(i) = \sum_{t=1}^{T_i} P(k|\boldsymbol{y}_t^{(i)}, \Omega)(\boldsymbol{y}_t^{(i)} - \boldsymbol{m}_k) \ . \tag{7}$$

To facilitate i-vector clustering, we normalize each i-vector to have a unit norm.

2.3. Hyperparameter Estimation

Given the training data $\mathcal{Y}$ and the pre-trained UBM-GMM Ω, the hyperparameters (i.e., total variability matrix) $\boldsymbol{T}$ can be estimated by maximizing the following objective function:

$$\mathcal{F}(\boldsymbol{T}) = \prod_{i=1}^{I} \int p(\boldsymbol{Y}_i|\boldsymbol{M}(i)) p(\boldsymbol{M}(i)|\boldsymbol{T}) \mathrm{d}\boldsymbol{M}(i). \tag{8}$$

Although it is possible to use variational Bayesian approach to solve the above problem, for simplicity, we use the following approximation to ease the problem:

$$p(\boldsymbol{Y}_i|\boldsymbol{M}(i)) \simeq \prod_{t=1}^{T_i} \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{y}_t^{(i)}; \boldsymbol{M}_k(i), \boldsymbol{R}_k)^{P(k|\boldsymbol{y}_t^{(i)}, \Omega)}.$$

Consequently, an EM-like algorithm can be used to solve the above simplified problem. The procedure for estimating $\boldsymbol{T}$ is described as follows:

Step 1: *Initialization*

Set the initial value of each element in $\boldsymbol{T}$ randomly from $[Th_1, Th_2]$, where Th_1 and Th_2 are two control parameters ($Th_1 = 0, Th_2 = 0.01$ in our experiments). For each training utterance, calculate the corresponding "Baum-Welch" statistics as in Eq. (6) and Eq. (7).

Step 2: *E-step*

For each training utterance $\boldsymbol{Y}_i$, calculate the posterior expectation of $\boldsymbol{w}(i)$ using the sufficient statistics and the current estimation of $\boldsymbol{T}$ as follows:

$$\begin{aligned} E[\boldsymbol{w}(i)] &= \boldsymbol{l}^{-1}(i) \boldsymbol{T}^\top \boldsymbol{R}_0{}^{-1} \boldsymbol{\Gamma}_{\boldsymbol{y}}(i) \\ E[\boldsymbol{w}(i)\boldsymbol{w}^\top(i)] &= E[\boldsymbol{w}(i)] E[\boldsymbol{w}^\top(i)] + \boldsymbol{l}^{-1}(i) \end{aligned} \tag{9}$$

where $\boldsymbol{l}(i)$ is defined in Eq. (5).

Step 3: *M-step*

Solve the following equation to update $\boldsymbol{T}$:

$$\sum_{i=1}^{I} \boldsymbol{\Gamma}(i) \boldsymbol{T} E[\boldsymbol{w}(i)\boldsymbol{w}^\top(i)] = \sum_{i=1}^{I} \boldsymbol{\Gamma}_{\boldsymbol{y}}(i) E[\boldsymbol{w}^\top(i)]. \tag{10}$$

Step 4: *Repeat or stop*

Repeat **Step 2** to **Step 3** for a fixed number of iterations or until the objective function in Eq. (8) converges.

3. i-Vector based Data Clustering

3.1. Clustering of i-Vectors using LBG Algorithm

As described above, given the training corpus, a unit-norm i-vector can be extracted from each training utterance. Given the set of training i-vectors, we use a hierarchical divisive clustering algorithm, namely LBG algorithm [4], to cluster them into multiple clusters. To measure the similarity between two i-vectors, $\hat{\boldsymbol{w}}(i)$ and $\hat{\boldsymbol{w}}(j)$, the following cosine similarity measure is used:

$$sim(\hat{\boldsymbol{w}}(i), \hat{\boldsymbol{w}}(j)) = \hat{\boldsymbol{w}}(i)^\top \hat{\boldsymbol{w}}(j). \tag{11}$$

Given the above similarity measure, it can be proven that the centroid, cw, of a cluster consisting of n unit-norm vectors, $\hat{\boldsymbol{w}}(1), \hat{\boldsymbol{w}}(2), \ldots, \hat{\boldsymbol{w}}(n)$, can be calculated as follows:

$$\begin{aligned} cw &= \operatorname*{argmax}_{cw} \sum_{i=1}^{n} sim(\hat{\boldsymbol{w}}(i), cw) \\ &= \begin{cases} \frac{\sum_{i=1}^{n} \hat{\boldsymbol{w}}(i)}{||\sum_{i=1}^{n} \hat{\boldsymbol{w}}(i)||} & \text{if } \sum_{i=1}^{n} \hat{\boldsymbol{w}}(i) \neq 0 \\ 0 & \text{otherwise} \end{cases} . \end{aligned} \tag{12}$$

After the convergence of the LBG clustering algorithm, we obtain E clusters of i-vectors with their centroids denoted as $cw_1, cw_2, \ldots, cw_E$, respectively. We use cw_0 to denote the centroid of all the training i-vectors.

3.2. Training Multiple Acoustic Models

Given the i-vector clustering result, each training utterance can be classified into one of E clusters. For each cluster, a cluster-dependent (CD) acoustic model (AM) can then be trained by using a cluster-independent (CI) AM as a seed. Consequently, we will have E CD AMs and one CI AM. Such trained multiple AMs can then be used in recognition stage to improve recognition accuracy.

3.3. Using Multiple Acoustic Models in Recognition

Apparently, there are many possible ways to use multiple AMs. In this study, we compare the following five methods:

- **Method 1:** Given an unknown utterance $\boldsymbol{Y}$, a unit-norm i-vector $\hat{\boldsymbol{w}}$ is extracted first. $\boldsymbol{Y}$ is then classified to a cluster, e, as follows:

  $$e = \operatorname*{argmax}_{l=1,2,\ldots,E} sim(\hat{\boldsymbol{w}}, cw_l).$$

  The CD AM of the e-th cluster will then be used to recognize $\boldsymbol{Y}$. This is the most efficient way to use multiple CD AMs.

Table 1: Comparison of five methods of using multiple acoustic models to improve recognition accuracy. The word error rate (WER) of the baseline system using cluster-independent model is 42.2%. Each cell shows the WER (relative WER reduction against the baseline) of the corresponding system, both in %.

# of Clusters	Method 1	Method 2	Method 3	Method 4	Method 5
2	41.6 (1.4)	41.2 (2.4)	41.2 (2.4)	41.3 (2.1)	41.1 (2.6)
4	41.4 (1.9)	40.7 (3.6)	40.6 (3.8)	40.3 (4.5)	40.1 (5.0)
8	41.2 (2.4)	40.5 (4.0)	40.4 (4.3)	40.0 (5.2)	39.7 (5.9)
16	40.9 (3.1)	40.3 (4.5)	39.9 (5.5)	39.4 (6.6)	39.4 (6.6)

- **Method 2:** Given an unknown utterance $\boldsymbol{Y}$, do i-vector based cluster selection as in **Method 1**. $\boldsymbol{Y}$ will be recognized by using both the selected CD AM and the CI AM via "parallel decoding". The final recognition result will be the one with a higher likelihood score. This method is computationally more expensive than **Method 1**.
- **Method 3:** Given an unknown utterance $\boldsymbol{Y}$, do i-vector based cluster selection by comparing $\hat{\boldsymbol{w}}$ with $E+1$ centroids, namely $cw_0, cw_1, \ldots, cw_E$, to identify top 2 most similar clusters. $\boldsymbol{Y}$ will then be recognized by using the two selected (CD and/or CI) AMs via "parallel decoding". This method has a similar run-time cost as **Method 2**.
- **Method 4:** Given an unknown utterance $\boldsymbol{Y}$, do "parallel decoding" by using E CD AMs and select the final recognition result with the highest likelihood score. This method has a much higher run-time cost than the previous three methods.
- **Method 5:** Given an unknown utterance $\boldsymbol{Y}$, do "parallel decoding" by using E CD AMs and one CI AM, and select the final recognition result with the highest likelihood score. This method has a similar run-time cost as **Method 4**.

4. Experiments and Results

4.1. Experimental Setup

As in [1], we also choose a Voice Search task to evaluate the effectiveness of our proposed approach. Our training data set consists of about 7,500 hours of narrow-band (8 KHz sampling rate) speech data (about 9M training utterances). About half of the training corpus was manually transcribed, including connected digits, read sentences, broadcast news, conversational telephony speech, etc., which are typically used by ASR research community to build different large vocabulary continuous speech recognition (LVCSR) systems. Another half was collected from Windows Live voice search service, and the word-level transcription accuracy was about 85%. As for testing data, we use 4,726 real-world voice search queries.

For feature extraction in front-end, we used 36 HLDA transformed features from 52 MFCC and its derivative features (i.e., $D = 36$). For acoustic modeling, we used phonetic decision-tree based tied-state triphone GMM-HMMs with 9,000 states and 48 Gaussian components per state. Our recognition vocabulary contains about 100K unique words and about 130K unique pronunciations. All of the recognition experiments were performed using the HDecode engine of HTK3.4 toolkit [5] with a trigram language model. Recognition performance was measured by the metric of word error rate (WER).

As for HMM training, only ML training was performed. The cluster-independent (CI) model was trained using the full training set. The cluster-dependent (CD) model was initialized using the CI model. Four ML iterations were performed to re-estimate the HMM model parameters (including means, variances, mixture component weights and state transition probabilities) using the cluster-specific training data. All the CD models shared the same model topology and the phonetic decision tree with the CI model.

Table 2: Distribution of word error rate (WER in %) (relative WER reduction in % against the baseline) of the **Method 1** across different subsets of testing sentences selected by different acoustic models (AMs). The number of CD AMs is 16.

Cluster ID	# of sentences	Baseline	Method 1
1	9	23.3%	26.7% (-14.3%)
2	145	57.8%	53.4% (7.66%)
3	27	30.7%	28.7% (6.5%)
4	189	64.1%	54.9% (14.4%)
5	141	38.1%	37.9% (0.6%)
6	6	46.2%	46.2% (0.0%)
7	339	41.7%	39.2% (5.9%)
8	268	41.2%	40.2% (2.6%)
9	558	37.4%	36.5% (2.4%)
10	1861	42.8%	41.9% (2.1%)
11	304	40.9%	39.8% (2.6%)
12	218	31.6%	31.1% (1.6%)
13	161	38.4%	37.6% (2.0%)
14	253	37.2%	36.4% (2.3%)
15	211	50.3%	52.1% (-3.6%)
16	36	52.1%	54.8% (-5.3%)
Total	4726	42.2%	40.9% (3.1%)

In i-vector extraction, the number of Gaussian components in UBM-GMM is $K = 1024$, and the dimension of i-vector is $F = 400$.

To handle the large-scale training data, the i-vector extraction and model training tools were implemented based upon MSR Asia's HPC-based speech training platform. This training platform was developed on top of Microsoft Windows HPC Server, and optimized for various speech training and other machine learning algorithms. With this high-performance parallel computing platform, we can run experiments very efficiently for such a large-scale task.

4.2. Experimental Results

Table 1 summarizes a comparison of five methods of using multiple acoustic models to improve recognition accuracy. The WER of the baseline system using cluster-independent model is 42.2%. Each cell in Table 1 shows the WER (relative WER reduction against the baseline) of the corresponding system, both in %. It is observed that although **Method 1** is most efficient at run-time, unfortunately it is less effective than **Method 4**, where expensive parallel decoding using multiple AMs is performed. By comparing **Method 2** and **Method 3** with **Method 1**, and comparing **Method 5** with **Method 4**, it is observed that incorporating CI AM is helpful. Overall, **Method 3** offers the best

Table 3: Distribution of word error rate (WER in %) (relative WER reduction in % against the baseline) of the **Method 4** across different subsets of testing sentences selected by different acoustic models (AMs). The number of CD AMs is 16.

Cluster ID	# of sentences	Baseline	Method 4
1	34	32.9%	25.0% (24.0%)
2	140	52.3%	43.8% (16.3%)
3	48	28.1%	26.1% (7.0%)
4	196	62.8%	52.5% (16.4%)
5	52	58.2%	51.6% (11.3%)
6	8	88.2%	82.4% (6.7%)
7	281	35.8%	33.9% (5.4%)
8	324	38.6%	34.6% (10.5%)
9	682	38.8%	37.0% (4.5%)
10	1628	41.6%	39.5% (5.1%)
11	182	46.1%	42.5% (7.6%)
12	224	31.8%	29.6% (6.9%)
13	136	46.9%	43.8% (6.6%)
14	417	35.6%	34.8% (2.1%)
15	366	60.2%	59.1% (1.9%)
16	8	68.8%	39.4% (22.3%)
Total	4726	42.2%	39.4% (6.6%)

tradeoff between efficiency and effectiveness.

Table 2 to Table 5 give detailed distributions of WER in % (relative WER reduction in % against the baseline) of the **Method 1**, **Method 4**, **Method 3**, **Method 5**, respectively, across different subsets of testing sentences selected by different acoustic models (AMs). The number of CD AMs is 16. In Table 4 and Table 5, Cluster ID number 0 corresponds to CI AM. It is quite clear that the relative WER reduction varies in a wide range for different subsets of testing data. This clearly shows the potential for further improvement. One known issue is the negative effect of non-speech portions on the result of i-vector extraction. Using an efficient VAD to remove non-speech portions before i-vector extraction will definitely be helpful. This is especially true for voice search data where long non-speech portions indeed exist at the beginning and end of the utterance, and between spoken keywords.

5. Conclusion and Discussion

We have presented an i-vector based approach to clustering training data for training multiple acoustic models to improve speech recognition accuracy. The proposed data clustering approach is very efficient therefore can deal with very large scale corpus. As future work, we will use a much larger and more representative testing set to study the effectiveness of the proposed approach. A more in-depth analysis of the results on such testing data will help us gain more insights so that a better strategy may be identified. Although our i-vector based data clustering approach is more efficient than the one reported in [1], it would be interesting to compare the effectiveness of these two approaches as another possible future work.

6. Acknowledgement

We thank our colleagues, Kaisheng Yao for providing us the baseline system and Yifan Gong for his support on this project.

7. References

[1] F. Beaufays, V. Vanhoucke, and B. Strope, "Unsupervised discovery and training of maximally dissimilar cluster models," *Proc. Interspeech-2010*, pp.66-69.

[2] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. on Audio, Speech and Language Processing*, Vol. 19, No. 4, pp.788-798, 2011.

[3] P. Kenny, G. Boulianne, P. Dumouchel, "Eigenvoice modeling with sparse training data," *IEEE Trans. on Speech and Audio Processing*, Vol. 13, No. 3, pp.345-354, 2005.

[4] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. on Communication*, Vol. COM-28, pp.84-95, 1980.

[5] S. Young, *et al.*, The HTK Book (for HTK version 3.4), 2006.

Table 4: Distribution of word error rate (WER in %) (relative WER reduction in % against the baseline) of the **Method 3** across different subsets of testing sentences selected by different acoustic models (AMs). The number of CD AMs is 16. Cluster ID number 0 corresponds to CI AM.

Cluster ID	# of sentences	Baseline	Method 3
0	100	40.4%	40.4% (0.0%)
1	11	31.3%	21.9% (30.0%)
2	148	57.6%	50.5% (12.4%)
3	32	24.4%	24.4% (0.0%)
4	190	63.9%	53.3% (16.6%)
5	69	42.9%	41.9% (2.4%)
6	3	44.4%	33.3% (25.0%)
7	325	38.6%	36.2% (6.4%)
8	289	39.5%	37.8% (4.3%)
9	611	38.9%	37.3% (4.1%)
10	1756	41.4%	39.3% (4.9%)
11	227	42.8%	41.7% (2.7%)
12	216	32.8%	31.5% (4.0%)
13	158	41.9%	41.7% (0.5%)
14	317	40.6%	40.3% (0.7%)
15	256	55.9%	52.8% (5.5%)
16	18	52.9%	61.8% (-16.7%)
Total	4726	42.2%	39.9% (5.5%)

Table 5: Distribution of word error rate (WER in %) (relative WER reduction in % against the baseline) of the **Method 5** across different subsets of testing sentences selected by different acoustic models (AMs). The number of CD AMs is 16. Cluster ID number 0 corresponds to CI AM.

Cluster ID	# of sentences	Baseline	Method 5
0	195	37.2%	37.2% (0.0%)
1	32	31.0%	22.5% (27.3%)
2	122	53.9%	44.7% (17.0%)
3	32	30.2%	28.1% (6.9%)
4	191	63.8%	53.1% (16.8%)
5	47	59.5%	54.1% (9.1%)
6	8	88.2%	82.4% (6.7%)
7	274	36.0%	33.8% (6.1%)
8	305	39.5%	35.4% (10.4%)
9	647	38.9%	37.1% (4.6%)
10	1590	41.6%	39.4% (5.2%)
11	172	45.8%	41.6% (9.2%)
12	217	31.7%	29.6% (6.8%)
13	133	47.5%	43.6% (8.2%)
14	404	35.9%	35.0% (2.5%)
15	350	60.6%	59.6% (1.7%)
16	7	71.4%	57.1% (20.0%)
Total	4726	42.2%	39.4% (6.6%)