Insight DCU at TRECVID 2015

Kevin McGuinness[1], Eva Mohedano[1], Amaia Salvador[3], Zhenxing Zhang[1], Mark Marsden[1], Peng Wang[4], Iveel Jargalsaikhan[1], Joseph Antony[1], Xavier Giro-i-Nieto[2], Shin'ichi Satoh[3], Noel E. O'Connor[1] Alan Smeaton[1]

[1]Insight Centre for Data Analytics, Dublin City University

[2]Universitat Politcnica de Catalunya, Barcelona

[3]National Institute of Informatics, Tokyo

[4]Tsinghua University, Beijing

Insight-DCU participated in the instance search (INS), semantic indexing (SIN), and localization tasks (LOC) this year. In the INS task we used deep convolutional network features trained on external data and the query data for this year to train our system. We submitted four runs, three based on convolutional network features, and one based on SIFT/BoW. `F_A_insightdcu_1` was an automatic run using features from the last convolutional layer of a deep network with bag-of-words encoding and achieved 0.123 mAP. `F_A_insightdcu_2` modified the previous run to use re-ranking based on an R-CNN model and achieved 0.111 mAP. `I_A_insightdcu_3`, our interactive run, achieved 0.269 mAP. Our SIFT-based run `F_A_insightdcu_2` used weak geometric consistency to improve performance over the previous year to 0.187 mAP. Overall we found that using features from the convolutional layers improved performance over features from the fully connected layers used in previous years, and that weak geometric consistency improves performance for local feature ranking. In the SIN task we again used convolutional network features, this time fine-tuning a network pretrained on external data for the task. We submitted four runs, `2C_D_A_insightdcu.15_1..4` varying the top-level learning algorithm and use of concept co-occurance. `2C_D_A_insightdcu.15_1` used a linear SVM top-level learner, and achieved 0.63 mAP. Exploiting concept co-occurance

improved the accuracy of our logistic regression run `2C_D_A_insightdcu.15_3` from 0.058 mAP to 0.6 `2C_D_A_insightdcu.15_3`. Our LOC system used training data from IACC.1.B and features similar to our INS run, but using a VLAD encoding instead of a bag-of-words. Unfortunately there was problem with the run that we are still investigating.

1 Introduction

Insight-DCU participated in three tasks for TRECVid 2015 [1]: instance search (INS), semantic indexing (SIN), and localization (LOC). We used features based on deep convolutional networks in all three tasks.

We used two distinct approaches in our INS submission. The first used deep convolutional networks to extract features, but, unlike our previous year's work, switched to using features from the convolutional layers instead of the fully connected layers, treating these as local features and encoding them using a bag-of-words model. We used a sparse matrix based inverted index and GPU-based sparse matrix multiplication to allow image query and ranking in fractions of a second. We participated in both the automatic and interactive search sub-tasks, doubling our mAP score in both tasks over last years result. Our interactive run ranked well above the overall median, and in 3rd place out of all interactive runs.

Our other INS submission used a classic SIFT bag-of-words model, but incorporated a form of weak geometric verification to perform fast re-ranking of results. Again, this improved performance significantly over our previous years submission.

Our SIN submission also used deep convolutional neural network based features. The network used to extract the features was based on the Oxford VGG-16 network, modified to have a 60 dimensional top layer and fine tuned on the 60 TRECVid concepts. Again the network improved our performance over last year, but failed to improve on our best progress task run. We also experimented with using concept co-occurances, with mixed results.

Like our instance search run, our LOC run used CNN features from the convolutional layers, but encoded these using VLAD. Unfortunately, however, there was some issues with our localization run that resulted in poor performance.

The remainder of this document describes the approaches and runs in more detail.

2 Instance Search

Two independent systems were developed for this submission, referred as *INS-1* and *INS-2*.

- The INS-1 system used a Bag of Visual Words (BoW) approach based on local CNN descriptors and was responsible of three of the runs submitted:
  1. `F_A_insightdcu_1`, mAP: 0.123
  2. `F_A_insightdcu_2`, mAP: 0.111
  3. `I_A_insightdcu_3`, mAP: 0.269
- The INS-2 system used an extension of our last year submission based on SIFT descriptors, including a new geometric validation technique for re-ranking:
  1. `F_A_insightdcu_4`, mAP: 0.187

2.1 INS-1 System

It has been shown that CNN image representations outperform handcrafted features in different image retrieval benchmarks [2]. Activations from the fully connected layers of a pre-trained CNN network designed for image classification can be used as a global image representation for retrieval [3, 4, 5]. However, those representations tend to be too high-level and and sometimes fail to capture sufficient local image detail for instance search. Recently, Razavian *et al* [2] showed than convolutional layers outperform fully connected ones in retrieval tasks and Yue-Hei *et al* [6] provide a guidance of how to extract local image representations from convolutional layers. For the INS-1 submissions, we designed a Bag of Visual Words system based on local CNN descriptors. The main difference with [6] is that we use a bag of words representation instead of VLAD to encode local information. This allows us to build an inverted index of the frames in the dataset and perform fast retrieval, which is especially important for practical interactive retrieval systems.

2.1.1 Dataset

Our dataset was created by performing uniform keyframe extraction at 1 frame-per-second, generating $\sim$ 1.5M images. Preliminary experiments were performed on a smaller dataset ($\sim$ 23,000 frames), referred as the *ground truth subset*, consisting of all relevant shots for the queries from TRECVid INS 2013.

Figure 1: Illustration of the spatial configuration of feature maps produced by the last convolutional layer. Each square is described by a 512D feature vector. Programme material © BBC.

2.1.2 Features

Feature extraction was performed using *Caffe* [7] and the pre-trained VGG-16 network [8]. All fully connected layers were removed from the network, making it possible to adapt the input size to our frames resolution (288×384). Activations from the last convolutional layer after max-pooling (layer *pool5*) were extracted, producing 9×12 feature maps of dimension 512 (Figure 1). Local descriptors were extracted by taking the activations of all the different filters for each coordinate in the feature maps, generating 108 local descriptors of dimension 512 per image.

2.1.3 Run 1: F_A_insightdcu_1

The run consisted of using local CNN features extracted from *pool5* as described in 2.1.2. A BoW model was built using a codebook of 100,000 visual words fit using k-means on a random subsample of 10 million local features. The BoW vectors were then transformed using *tf-idf* weighting and used to create an inverted index for the dataset.

At query time, CNN features from *pool5* were extracted for each topic image and mapped to an L_2 normalized bag-of-words representation using the codebook. The active words in the representation were then reweighted in inverse proportion to their distance from the object in the query image. In practice, this was done by filtering the binary mask

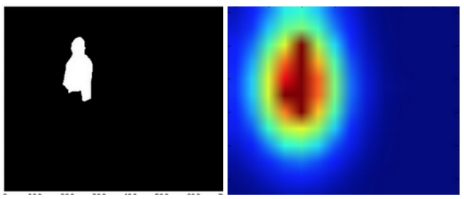

Figure 2: Example of a filtered binary mask with $\sigma = 100$.

for the query image with a large sigma Gaussian filter ($\sigma = 100$), resizing the result to match the size of the feature maps (9×12) and linearly re-normalizing the weights for each feature fill the range $[0, 1]$. The effect is to up-weight features from the foreground and assign small weights to background features. Figure 2 illustrates the process.

The final query vectors for each topic were then fused into a single representation and used to find images in the inverted index sharing at least one visual word with the topic images. The final result list was then ranked using a vector space model (cosine similarity).

2.1.4 Run 2: F_A_insightdcu_2

The top 1,000 results obtained with the method described in Section 2.1 were reranked using a state-of-the-art CNN trained for the task of object detection. We used the architecture and weights of Fast R-CNN [9] and fine tuned it for 31 classes: the 30 TRECVid INS queries and an extra class for the background.

The ground truth bounding boxes provided for the query images were used as training data. We further extended the training dataset using a fixed grid at different scales and aspect ratios over the image, selecting those windows that highly overlap with the ground truth bounding boxes as positives, and the rest as negatives. The IoU threshold was optimized using TRECVid INS 2014 data and fixed to 0.1. In total, each image was assigned an average of 6,000 boxes, resulting in 720,000 training examples to train the network. We trained for 40,000 iterations with a batch size of 128. The learning rate was set to 0.001, with a weight decay of 0.0005 after 30,000 iterations.

At test time, for each image we used the same grid to obtain the prediction for the 30 TRECVid classes (and the background class) at different image locations. We used the 31 activations of the last fully connected layer directly as the class probabilities for each region. Finally, for each class, the score assigned to a test image corresponds to the highest score obtained among all locations for that class (max-pooling).

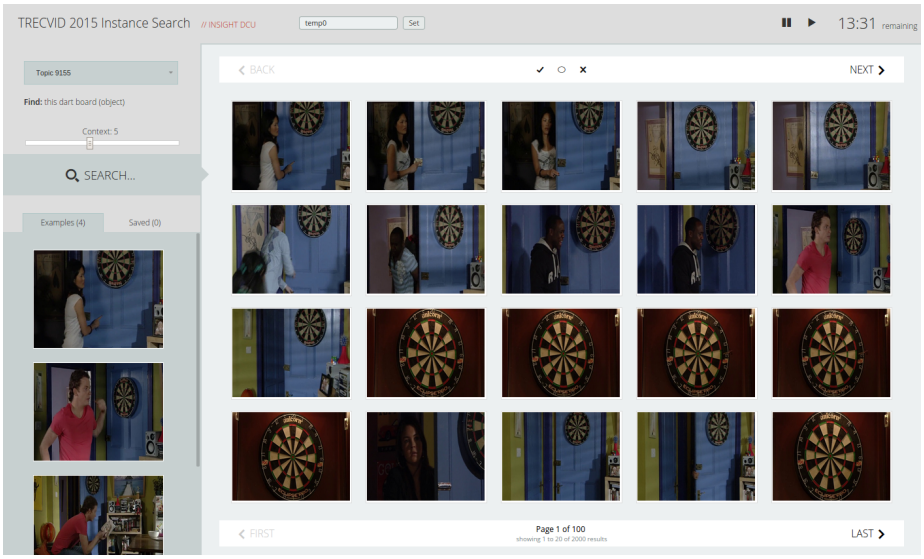

Figure 3: User interface for the interactive INS task. Programme material © BBC.

2.1.5 Run 3: I_A_insightdcu_3

User annotations were collected to improve the automatic results of the first submitted run. The interface (Figure 3) was extended from the one developed in our previous year's submission [10]. The new interface provides the capacity to select the relative influence of foreground and background features from the topic images using a slider. Adjusting the slider has an effect similar to altering the σ of the Gaussian used to weight the features in Section 2.1.3. The interface also allows users to search using individual topic images or using results retrieved from previous searches.

At query time, we used a sparse matrix representation of the inverted index and used a CUDA-based sparse matrix library[1] to perform inverted index lookups via sparse matrix vector products on a NVIDIA GTX Titan X GPU. The entire sparse matrix inverted index was loaded onto the GPU in advance, allowing searches to be performed over the entire dataset in fractions of a seconds.

Users were given 15 minutes to annotate as many positive and negative results as possible for each topic. With the collected annotations, a reranking of the top 1,000 retrieved shots was performed by putting all positively annotated results on top of the list, removing all results with negative annotations, and filling the remainder of the list with the top 1,000

[1] cuSPARSE: https://developer.nvidia.com/cusparse

results obtained from the original automatic list from Section 2.1.3. This simple strategy increased mAP from 0.123 (run 1) to 0.269.

2.1.6 Results and Discussion

Local CNN features allowed us to adapt the classical BoW approach with learned features. By using this approach, we considerably improved our performance over last year result, which was based on features from fully-connected layers. We have improved the automatic run from 0.062 mAP (INS-2014 automatic approach) to 0.123 mAP. We have also doubled the performance of the interactive run with respect to last year results: from 0.135 mAP (our best INS-2014 interactive submission) to 0.269 mAP. The new approach ranks 3rd in the interactive runs and performs well above the interactive median (0.17) and overall median (0.18).

These results are evidence that lower layers contain local information that is more useful for instance search, when compared to fully connected layers, whose information is too high-level for this task. Future work will include spatial verification of our local features to improve the automatic rankings and more sophisticated use of the manual annotations to improve the results.

2.2 INS-2 System

In this years experiment, our instance search run 4 used a spatial verification approach [11] to improve the efficiency and effectiveness by boosting the retrieval performance with geometric information from objects. Unlike existing methods that estimate a full transformation model from matched feature points, we consider the geometric correlation among local features and hypothesize that the pairwise geometric correlation between consistent matches should also be consistent and follow the same spatial transformation between objects. We explored and incorporated these correlations to measure transformation consistency in rotation and scale space and used it to effectively eliminate inconsistent feature matches at a low-computational cost. The approach can therefore be applied to all candidate images (instead of just a subset).

2.2.1 Motivation

In a standard BoVW architecture the SIFT [12] algorithm is used to extract the local features from each image to encode invariant visual information in these feature vectors. Normally a feature vector is defined as $\vec{v} = (x, y, \theta, \sigma, q)$, where variables $\{x, y, \theta, \sigma\}$ stand for the local salient point's 2-D spatial location, dominant orientation, and most stable

scale, respectively. For a given list of candidate images, the task of spatial verification is to eliminate the unreliable feature matches and only retain the match set C_{stable} that links the patches from the same objects. I.e.:

$$C_{stable} = \{m_i \in C_{initial} \quad \text{and} \quad f_{sp}(m_i) = 1\}, \tag{1}$$

where m_i stands for the i^{th} feature match from the initial match set between a query image I_q and candidate image I_c, and f_{sp} is a binary spatial verification function for assessing geometric consistency. Instead of verifying the geometric consistency for each match individually, we propose a novel approach to verify the consistency between pairwise geometric correlations along with their corresponding feature points. For a given pair of feature matches m_l and m_n, we define the spatial verification function as:

$$f_{sp} = \begin{cases} 1 & \text{if } \Delta\theta, \Delta\theta_{l\to n} \in D_\theta \text{ and } \Delta\sigma, \Delta\sigma_{l\to n} \in D_\sigma \\ 0 & \text{if otherwise,} \end{cases} \tag{2}$$

where $\Delta\theta$ and $\Delta\sigma$ verified the consistency of geometric transformation for each individual feature match against the dominant transformation D_θ and D_σ in orientation and scale space. The geometric correlation from feature match m_l to m_n, represented by $\Delta\theta_{l\to n}$ and $\Delta\sigma_{l\to n}$, are also verified to further improve the performance of spatial verification. Compared to the state-of-the-art spatial verification approaches, which are required to estimate a 3-D transformation model for verification, our approach estimates weak consistencies and applies the verification at a much lower computational cost.

Figure 4 demonstrates our idea of using geometric correlations to assess reliability of feature matches. The object of interest (front cover of a box) is highlighted with dark yellow box. Matches (A, A'), (B, B') are considered to be consistent because the spatial transformation is consistent between match (A, A'), (B, B'), and their correlation $(AB, A'B')$. On the other hand, match (C, C') is filtered out due to the fact that geometric correlation between $(AC, A'C')$ is not consistent with the spatial transformation.

2.2.2 Experiment and Results

We developed the complete instance search system from our previous participation [10] and integrated the weak geometric correlation consistency approach to further improve retrieval performance.

To explicitly examine all the correlations between the initial feature matches is a non-trivial problem. In this work, we proposed a three-step scheme to reduce the complexity of verifying the geometric correlation consistency, and to make it applicable at low cost for large-scale instance search systems. The approach used was as follows: *a*) first, we establish a weak geometric transformation, specifically in rotation and scaling space, from

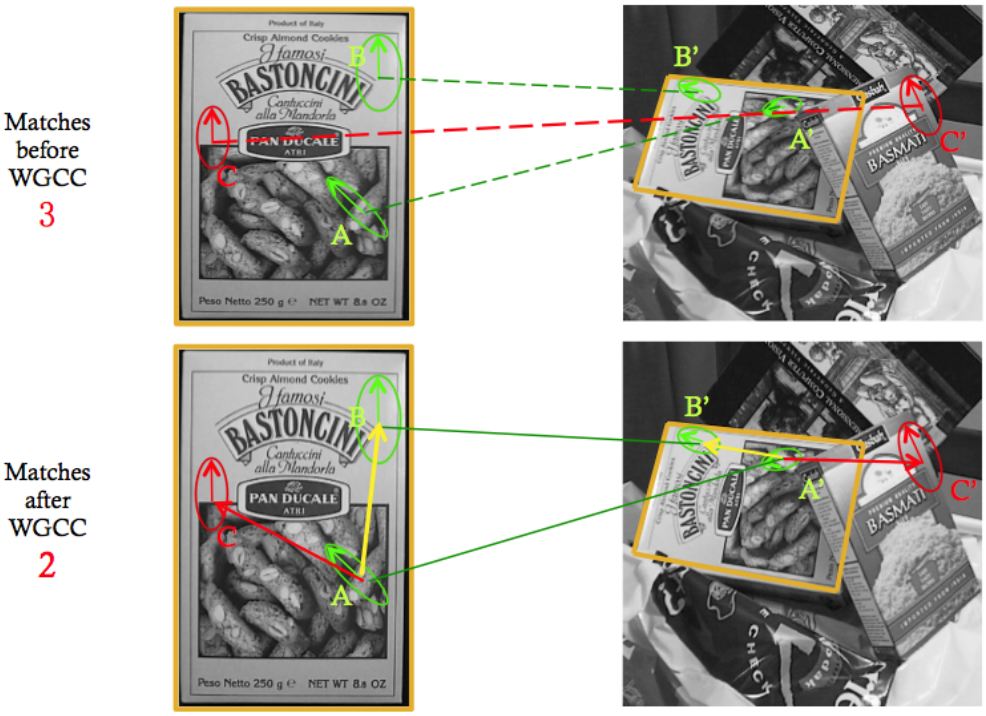

Figure 4: An illustration of verifying consistency of feature matches using Weak Geometric Correlations Consistency (WGCC). The green (red) line indicates the consistent (inconsistent) feature matches.

the initial set of feature matches with the Hough voting scheme; *b*) then we determine the strongest feature match that will be served as a reference match in the step of verifying geometric correlations; *c*) finally we identify the reliable feature matches by verifying the consistency of the geometric correlations from each feature match to the reference matches. The computation cost are closely related to the total number of initial feature matches. We reduced the major computational cost by adopting a topology-based graph match to identifying the reference matches in the proposed scheme.

We used late fusion to make use of multiple query images and the evaluation results suggested that our proposed approach could improve the precision accuracy in instance search tasks. For our automatic run, the system performance improved by about 64% in from 0.12 mAP last year to 0.187 this year. In particular, our system achieves over 0.7 mAP for topic 9153, 9157, and 9158, which strongly demonstrates the success of our

approach.

3 Semantic Indexing

Insight-DCU submitted four runs for the TRECVid 2015 SIN (semantic indexing) task, with two previously submitted runs also evaluated as part of the progress task.

- **Main Task** :

  `2C_M_D_insightdcu.15_1, 2C_M_D_insightdcu.15_2,`

  `2C_M_D_insightdcu.15_3, 2C_M_D_insightdcu.15_4.`

- **Progress Task** :

  `2C_M_A_insightdcu.13_1, 2C_M_A_insightdcu.14_1.`

3.1 Datasets

The development dataset used for the main and progress tasks combines the development and test datasets of the 2010 and 2011 SIN tasks: IACC.1.tv10.training, IACC.1.A, IACC.1.B, and IACC.1.C. For the 2015 submissions, this development set was indirectly combined with non-TRECVid training data through the use of a pretrained convolutional network model.

3.2 Features

Deep convolutional neural network (CNN) features were used for this year's SIN task. Retraining an existing deep learning model for a specific classification problem has been shown to achieve strong benchmarking performance at a lower computational cost than training from scratch [13]. A high performing CNN model [8] was retrained for the 2015 SIN task using the key frames supplied with the development dataset and the Caffe framework [7]. The final fully connected layer of the chosen network [8] was altered to have a 60D output in an attempt to avoid overfitting (this also matches the number of concepts required to be recognised for the SIN task). Stochastic gradient descent was then performed on this network for 54,000 iterations, with 85% of the development dataset keyframes used as a training set and the remaining 15% used for validation. The resulting network was then used to produce a task specific, 60D image descriptor through forward propagation.

Run Name	MinfAP (%)
2C_M_A_insightdcu.13_1	9.8
2C_M_A_insightdcu.14_1	3.8
2C_D_A_insightdcu.15_1	6.3
2C_D_A_insightdcu.15_2	4.8
2C_D_A_insightdcu.15_3	5.8
2C_D_A_insightdcu.15_4	6.0

Table 1: MinfAP scores for 2015 semantic indexing 2015

3.3 Ranking Shots for Concept Occurrence

The 60D image descriptor was calculated for all training set keyframes and then used to train a layer of 60 one-v-all linear support vector machines (SVM). Each SVM produces a hyperplane distance score for a given semantic concept for a given image. Using the test set keyframes (IACC.2.C), a shot ranking is generated for each semantic concept using scores produced by the layer of SVMs. This set of rankings forms our baseline submission (`2C_M_D_insightdcu.15_1`).

3.4 Concept Co-Occurrence Processing

Semantic concepts found in visual media tend to occur in groups, with both likely and unlikely combinations identifiable [14]. A training-free refinement (TFR) algorithm for enhancing the semantic indexing of visual media based purely on concept detection results was used to enhance our baseline run [14]. This method combines the correlation of individual concepts with various detection accuracy values in order to refine the rankings and improve the overall semantic indexing performance. The SVM hyperplane distance values produced by our baseline run were normalised between the max and min value's recorded for each concept and then processed using this TFR algorithm to produce a new set of rankings (`2C_M_D_insightdcu.15_2`).Though usable in this manner, normalised hyperplane distances are not ideal for this kind of processing as the algorithm assumes genuine probability values are being supplied. To remedy this the SVM layer was replaced with a logistic regression layer to produce a new set of probability based rankings (`2C_M_D_insightdcu.15_3`). Following this concept co-occurrence processing was again performed, producing our final run (`2C_M_D_insightdcu.15_4`).

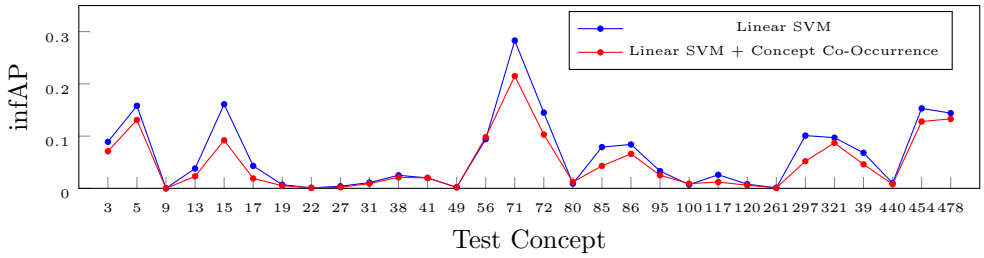

Figure 5: InfAP scores for `2C_M_D_insightdcu.15_1` and `2C_M_D_insightdcu.15_2`

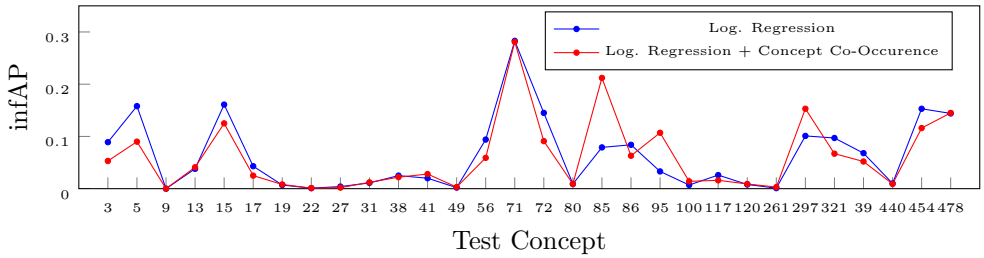

Figure 6: InfAP scores for `2C_M_D_insightdcu.15_3` and `2C_M_D_insightdcu.15_4`

3.5 Results and Discussion

The classification performance of the six (four main and two progress) runs was evaluated using inferred average precision (infAP). Table 1 shows the mean infAP results for all six runs while figures 5 and 6 show the per-concept infAP scores for our SVM and logistic regression based runs respectively.

Concept co-occurrence processing is seen to improve the infAP of our logistic regression based run and harm that of our our SVM based run. This is not unexpected given that normalised hyperplane distance values are not true probability values. The concept for which the best overall performance is achieved across the 2015 runs is "Instrumental_Musician." The best 2015 run (`2C_M_D_insightdcu.15_1`) achieves greater performance (MinfAP= 6.3%) than the 2014 progress run (MinfAP= 3.8%) but fails to outperform the 2013 progress run ((MinfAP= 9.8%). Further experiments could involve denser keyframe sampling and the performance of concept co-occurrence processing for the full 500 SIN concepts before ranking the chosen 60, which should allow the TFR algorithm to make an even greater improvement in terms of infAP.

4 Localization

Insight-DCU has submitted single run for the TRECVid 2015 LOC (Localization) task.

- **Localization Task** :

  `DCU_Localization_First_Run_Results`

The goal is to detect concepts precisely in time and space from videos.

4.1 Datasets

The dataset (IACC.2. A, B, and C) consist of approximately 7300 internet video archives (600 hours, 144 GB). The A and B sets are used for training and the localisation results should be submitted for set C. The keyframes (I-frames) for all the 3 sets were available.

Ground truth: the available key frames included 1,566,568 key frames from set A, 1,573,832 from set B, and from 111,597 set C. Though it is mentioned in the guidelines to use A set and B set for training, we found ground truth only for a limited images from 2013 dataset. For a set of key frames in the training set B (approximately 62,500), we found the ground truth (I-frames with labels and bounding box coordinates) for 8 concepts excluding "Anchor person" (5) and "Computers" (31), out of the concepts for this year.

4.2 Deep Learning Features

We have used the VGG ILSVRC convolutional neural network with 16 layers, the same used for INS task this year. After resizing all the input images to 240×240 pixels and we extracted the features from the *pool5* layer. The dimension of the features were 64×512 for each image.

4.2.1 Feature Representation

After generating a codebook (vocabulary) from the extracted CNN features, two methods were used for feature representations: namely *Bag-of-Visual-Words* (BoW) and *VLAD*. For the evaluation, we have only submitted the *VLAD* based method as it has showed better performance compared to Bow.

For the BoW method, the codebook dictionary was fit using the k-means algorithm. We evaluated the codebook sizes of $k = 50$ and $k = 100$ centroids fit using $100,000$ randomly sampled features from the training set. The resulting representation was L_2 normalized.

Method	Avg. F-Score
BoW	0.09
VLAD ($k = 6$)	0.41
VLAD ($k = 10$)	0.42

Table 2: Evaluation on a 30/70 train/validation split of the training set.

For the VLAD representation, we set the number of centroids to be $k = 10$. The resulting VLAD vectors were normalized using local and global L_2 normalization. The VLAD representation gave better results than BoW, but result in a very high dimensional features. We therefore used less centroids to handle memory limitations.

4.3 Classifier Training

A multiclass support vector machine (SVM) with a linear kernel was used for training classifierS. We trained multiclass SVM classifiers for the BoW and VLAD representations. To validate of the techniques, we have evaluated on a split of 70% for training and 30% for testing. The validation performance is shown in Table 2. The VLAD representation achieved the significantly better than the BoW representation.

4.4 Results and Discussion

We submitted a single run (`DCU_Localization_First_Run_Results`) based on the $VLAD$ representation. Due to the time constraint, we could not provide the spatial localization of objects, but only temporal segmentation.

Table 3 shows the performance of the run. Unfortunately, the final results were not as expected: the accuracy was worse than a random classifier. We suspect this was due to a bug in our code but have not yet been able to establish the reason for the poor performance of the final submission. The method followed a standard standard object detection pipeline and gave reasonable performance in validation, as can be seen from Table 2. In the future, we are planning to test the same technique on standard object localization benchmark datasets and investigate our approach in detail.

Metric	Score
Avg. I-frame F-Score	0.055
Avg. I-frame Precision	0.096
Avg. I-frame Recall	0.064

Table 3: Final evaluation results

5 Conclusion

In instance search, we found that switching to the lower convolutional layers and encoding these using a bag-of-words model was significantly more effective than using features from the higher-level fully connected layers. We hypothesize that this is because the higher layers discard too much local detail, which is critical for instance search. Using a bag of words model also allowed us to perform very fast ranking, using sparse matrix multiplications on GPU hardware.

In the semantic indexing task we found that fine tuning the network improved our performance over previous years, but not dramatically so. Using concept co-occurrence gave mixed results – improving performance when the outputs were probabilities generated by logistic regression, and reducing it when the outputs were scaled hyperplane distances from a linear support vector machine.

Acknowledgments This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under grant number SFI/12/RC/2289. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the GPUs used for this research. Programme material for the instance search task is © BBC.

References

[1] Paul Over, George Awad, Martial Michel, Jonathan Fiscus, Wessel Kraaij, Alan F. Smeaton, Georges Quenot, and Roeland Ordelman. Trecvid 2015 – an overview of the goals, tasks, data, evaluation mechanisms and metrics. In *Proceedings of TRECVID 2015*. NIST, USA, 2015.

[2] A. Razavian, J Sullivan, A. Maki, and S. Carlsson. Visual instance retrieval with deep convolutional networks. *CoRR*, abs/1412.6574, 2014.

[3] J. Wan, D. Wang, S. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li. Deep learning for content-based image retrieval: A comprehensive study. In *Proceedings of the ACM*

International Conference on Multimedia, MM '14, pages 157–166, New York, NY, USA, 2014. ACM.

[4] Artem Babenko, Anton Slesarev, Alexandr Chigorin, and Victor Lempitsky. Neural codes for image retrieval. In *Computer Vision–ECCV 2014*, pages 584–599. 2014.

[5] A. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. *CoRR*, abs/1403.6382, 2014.

[6] Y. Yue-Hei Ng, F. Yang, and L. Davis. Exploiting local features from deep networks for image retrieval. *CoRR*, abs/1504.05133, 2015.

[7] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *Computer Vision and Pattern Recognition*, 2014.

[8] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[9] Ross Girshick. Fast R-CNN. *arXiv preprint arXiv:1504.08083*, 2015.

[10] K. McGuinness, E. Mohedano, Z. Zhang, F. Hu, R. Albatal, C. Gurrin, N. O'Connor, A. Smeaton, A. Salvador, X. Giró-i Nieto, and C. Ventura. Insight centre for data analytics (DCU) at TRECVid 2014: Instance search and semantic indexing tasks. In *2014 TRECVID Workshop*, Orlando, Florida (USA), 11/2014 2014. National Institute of Standards and Technology (NIST), National Institute of Standards and Technology (NIST).

[11] Zhenxing Zhang, Rami Albatal, Cathal Gurrin, and Alan F Smeaton. Instance search with weak geometric correlation consistency. In *The 22TH International Conference of MultiMedia Modeling*, Miami,USA, 2016.

[12] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004.

[13] Sergey Karayev, Matthew Trentacoste, Helen Han, Aseem Agarwala, Trevor Darrell, Aaron Hertzmann, and Holger Winnemoeller. Recognizing image style. *Computer Vision and Pattern Recognition*, 2013.

[14] Peng Wang, Alan F Smeaton, and Cathal Gurrin. Factorizing time-aware multi-way tensors for enhancing semantic wearable sensing. In *MultiMedia Modeling*, pages 571–582. Springer, 2015.