Automatic Image Annotation using Deep Learning Representations

`Image annotation/tagging`

Novel Ideas

1. **employ the Word2Vector to represent the textual tag.**
2. **present a simple CNN based linear regression model, no hand-crafted feature, less computationally cost.**

Previous Works

1. **Generative models**: work by computing the joint probability of words and visual features. During test steps, given a test image, the model is used to compute conditional probability scores for words.
   Such as CMRM(Cross Media Relevance Model), CRM(Continuous Relevance Model), MBRM(Multiple Bernouli Relevance Model)
2. **Discriminative models**:
3. **Nearest Neighbor models**: find semantic neighbors for each image. the tags are predicted based on the weighted combination of distances.
   such as **JEC(Joint Equal Contribution), TagProp, 2PKNN(2 Pass KNN)**
4. **CRM based models**: CRM using Sparse Kernel Learning (SKL-CRM)

Feature Extraction

1. **CNN Feature:** extract the fc-layer feature as 4096-dimensional vector in **VGG-19** pretrained on the ILSVRC-2012
2. **Word embeddings:** obtained from a pretrained skip-gram text modeling architecture.

3. **Kernel CCA:** use a χ^2 kernel for exploiting the non-linear relationships. $K_x = \langle \phi_x, \phi_x \rangle$, $K_y = \langle \phi_y, \phi_y \rangle$.

$$arg\ max_{\alpha,\beta} \frac{\alpha^T K_x K_y \beta}{\sqrt{(\alpha^T K_x^2 \alpha + r_x \alpha^T K_x \alpha)(\beta^T K_y^2 \beta + r_y \beta^T K_y \beta)}}$$

CNN Regression model

Formulate the annotation problem as a linear problem, we replace the final layer in the caffe model with a projection layer.
We use Euclidean Loss(L2) instead of Softmax Loss during the training phase.

Experiments

Method	Feature Visual	Feature text	Corel-5K P	Corel-5K R	Corel-5K F	Corel-5K N+	ESP Game P	ESP Game R	ESP Game F	ESP Game N+	IAPRTC-12 P	IAPRTC-12 R	IAPRTC-12 F	IAPRTC-12 N+
JEC [10]	HC	-	27	32	29	139	22	25	23	224	28	29	29	250
MBRM [2]	HC	-	24	25	25	122	18	19	19	209	24	23	24	223
TagProp(σML) [4]	HC	-	33	42	37	160	39	27	32	239	46	35	40	266
2PKNN [15]	HC	-	39	40	39.5	177	51	23	31.7	245	49	32	38.7	274
2PKNN+ML [15]	HC	-	44	46	45	191	53	27	36	252	54	37	**44**	278
SVM-DMBRM [13]	HC	-	36	48	41	197	**55**	25	34	259	56	29	38	**283**
KCCA-2PKNN [1]	HC	-	42	46	44	179	-	-	-	-	**59**	30	40	259
SKL-CRM [12]	HC	-	39	46	42	184	41	26	32	248	47	32	38	274
JEC	VGG-16	-	31	32	32	141	26	22	24	234	28	21	24	237
2PKNN	VGG-16	-	33	30	32	160	40	23	29	250	38	23	29	261
SVM-DMBRM	VGG-16	-	42	45	43	186	51	26	35	251	58	27	37	268
CCA	VGG-16	W2V	35	46	40	172	29	32	30	250	33	32	33	268
KCCA	VGG-16	W2V	39	**53**	45	184	30	36	33	252	38	**39**	38	273
CCA-KNN	VGG-16	BV	39	51	44	192	44	32	37	254	41	34	37	273
CCA-KNN	VGG-16	W2V	**42**	52	**46**	**201**	46	**36**	**41**	**260**	45	38	41	278
CNN-R	Caffe-Net	W2V	32	41.3	37.2	166	44.5	28.5	34.7	248	49	31	37.9	272

Some drawbacks

1. **The length of the annotations is fixed**
2. **It is a place in the performance showed by CNN-R**
3. **Since the size of dataset is not as much as, it is probably having a overfitting. The performance can be improved further with some regularization.**

得到tag的流程

1. 通过CCA或者KCCA的方法，训练得到映射矩阵W_x, W_y，W_x表示visual feature X的映射，W_y表示word embedding vector Y的映射，之所以使用CCA就是为了保持两者之间的相关性最大，这样映射后的两个变量之间的相关性最大

$$\rho = \arg\max_{\omega_x, \omega_y} \frac{\omega_x^T XY^T \omega_y}{\sqrt{(\omega_x^T XX^T \omega_x)(\omega_y^T YY^T \omega_y)}}$$

2. 当给一张test image，首先得到deep learning visual features V_t, 然后使用映射矩阵得到$T = (V_t - \mu_X)W_x$, 然后计算V_t 和 $V = (Y - \mu_Y)W_y$之间的相关性系数(注意此处的Y是tag的word embedding vector，所以Y有多个，而且不相同)。根据系数的大小和tag出现的frequency排序，来给image赋