

# Error evaluation for ImageCLEF 2009 IRMA Medical Image Annotation

This document explains how the errors will be evaluated for the ImageCLEF 2009 challenge, IRMA Medical Image Annotation task. Each image has four labels corresponding to the classification defined for the 2005, 2006, 2007 and 2008 editions of the challenge. Remember that the final ranking will be based on the sum of the error score evaluated on the four years classification outputs.

**2005 and 2006.** For these two years the error is evaluated just on the capability of the algorithm to make the correct decision. There is also the possibility to say “*don't know*”, which is encoded by \*.

For example:

correct:	18	
classified	error	count
18	0.0	
21	1.0	
*	0.5	

**2007 and 2008.** For these two years, the error is evaluated on the basis of the hierarchical IRMA code TTTT-DDD-AAA-BBB. The IRMA coding system consists of four independent axes with three to four positions, each in {0, ..., 9, a, ..., z}, where “0” denotes “unspecified” to determine the end of a path along an axis. The technical code (T) describes the imaging modality; the directional code (D) models body orientations; the anatomical code (A) refers to the body region examined; and the biological code (B) describes the biological system examined.

The code is strictly hierarchical – each sub-code element is connected to only one code element. The element to the right is a sub element of the element to the left.

For example:

```
4 upper extremity
41 upper extremity; hand
411 upper extremity; hand; finger
```

The finger is an element of the hand, which in turn is an element of the upper extremity.

Let an image be coded by the above 4 independent axes, such that we can consider the axes

separately and just sum up the errors for each axis independently.

Let  $l_1^I = l_1, l_2, \dots, l_i, \dots, l_I$  be the *correct* code (for one axis) of an image.

Let  $\hat{l}_1^I = \hat{l}_1, \hat{l}_2, \dots, \hat{l}_i, \dots, \hat{l}_I$  be the *classified* code (for one axis) of an image.

Where  $l_i$  is specified precisely for every position, and in  $\hat{l}_i$  is allowed to say “*don't know*”, which is encoded by \*. Note that  $I$  (the depth of the tree to which the classification is specified) may be different for different images.

Given an incorrect classification at position  $\hat{l}_i$  we consider all succeeding decisions to be wrong and given a not specified position, we consider all succeeding decisions to be not specified. Furthermore, we do not count any error if the correct code is unspecified and the predicted code is a wildcard. In that case, we do consider all remaining positions to be not specified.

We want to penalize wrong decisions that are easy (fewer possible choices at that node) over wrong decisions that are difficult (many possible choices at that node), we can say, a decision at position  $l_i$  is correct by chance with a probability of  $\frac{1}{b_i}$  if  $b_i$  is the number of possible labels for position  $i$ . This assumes equal priors for each class at each position.

Furthermore, we want to penalize wrong decisions at an early stage in the code (higher up in the hierarchy) over wrong decisions at a later stage in the code (lower down on the hierarchy) (i.e.  $l_i$  is more important than  $l_{i+1}$ ).

Putting together:

$$\sum_{i=1}^I \underbrace{\frac{1}{b_i}}_{(a)} \underbrace{\frac{1}{i}}_{(b)} \underbrace{\delta(l_i, \hat{l}_i)}_{(c)}$$

with

$$\delta(l_i, \hat{l}_i) = \begin{cases} 0 & \text{if } l_j = \hat{l}_j \quad \forall j \leq i \\ 0.5 & \text{if } l_j = * \quad \exists j \leq i \\ 1 & \text{if } l_j \neq \hat{l}_j \quad \exists j \leq i \end{cases}$$

where the parts of the equation are:

- (a) accounts for difficulty of the decision at position  $i$  (branching factor);

- (b) accounts for the level in the hierarchy (position in the string);
- (c) correct/not specified/wrong, respectively.

In addition, for every axis, the maximal possible error is calculated and the errors are normed such that a completely wrong decision (i.e. all positions for that axis wrong) gets an error count of 0.25 and a completely correctly predicted axis has an error of 0. Thus, an image where all positions in all axes are wrong has an error count of 1, and an image where all positions in all axes are correct has an error count of 0. Finally setting a wildcard \* instead of a 0 is not considered a mistake.

Examples:

```
third axis, correct: 463
classified error count
463          0.000000
46*          0.025531
461          0.051061
4*1          0.069297
4**          0.069297
47*          0.138594
473          0.138594
477          0.138594
***          0.125000
731          0.250000
```

**Clutter in 2005, 2006 and 2007.** For these three years we introduced a class called “clutter” C. Even if in the test set there will be images belonging to this class, their classification would NOT influence the error score for the challenge. The label assigned to those images will be used after the challenge for a more detailed analysis of the proposed algorithms.

Example 2005, 2006:

```
correct: C
classified error count
18          0.0
21          0.0
*           0.0
```

Example 2007:

```
third axis, correct: CCC
classified error count
111          0.000000
11*          0.000000
1**          0.000000
***          0.000000
*C*          0.000000
```