

detection\_utils.utils.object  
\_detection\_evaluation\_test.Weighted  
PascalEvaluationTest.test\_returns  
\_correct\_metric\_values

detection\_utils.utils.object  
\_detection\_evaluation\_test.Precision  
AtRecallEvaluationTest.test\_returns  
\_correct\_metric\_values

detection\_utils.utils.object  
\_detection\_evaluation\_test.Weighted  
PascalEvaluationTest.test\_returns  
\_correct\_metric\_values\_with\_difficult\_list

detection\_utils.utils.object  
\_detection\_evaluation\_test.Precision  
AtRecallEvaluationTest.test\_returns  
\_correct\_metric\_values\_with\_difficult\_list

detection\_utils.utils.object  
\_detection\_evaluation\_test.Weighted  
PascalEvaluationTest.create\_and  
\_add\_common\_ground\_truth

```
graph LR; A["detection_utils.utils.object  
_detection_evaluation_test.Weighted  
PascalEvaluationTest.test_returns  
_correct_metric_values"] --> D["detection_utils.utils.object  
_detection_evaluation_test.Weighted  
PascalEvaluationTest.create_and  
_add_common_ground_truth"]; B["detection_utils.utils.object  
_detection_evaluation_test.Precision  
AtRecallEvaluationTest.test_returns  
_correct_metric_values"] --> D; C["detection_utils.utils.object  
_detection_evaluation_test.Weighted  
PascalEvaluationTest.test_returns  
_correct_metric_values_with_difficult_list"] --> D; E["detection_utils.utils.object  
_detection_evaluation_test.Precision  
AtRecallEvaluationTest.test_returns  
_correct_metric_values_with_difficult_list"] --> D;
```