User Models, Metrics and Measures of Search: A Tutorial on the CWL Evaluation Framework ACM CHIIR UMMMS 2021

by
Leif Azzopardi, Alistair Moffat, Paul Thomas and Guido Zuccon

https://github.com/ireval/cwl

CWL-EVAL DEMO

cwl-eval demo

- How to install cwl-eval?
  - You will need Python 3.7
  - pip install cwl-eval

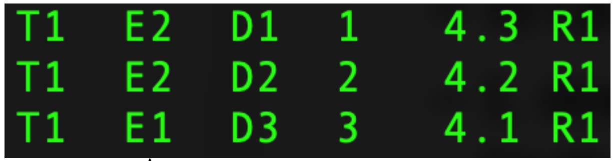
```
pip install cwl-eval
```

- How to use cwl-eval?
  - Cwl-eval takes the following Inputs:
    - gain_file (which is in trec qrel file format)
      - Binary relevance (1,0) can be used or
      - Gain values (typically between 0 and 1)
    - result_file (which is in trec result file format)

```
cwl-eval qrel_file result_file
```

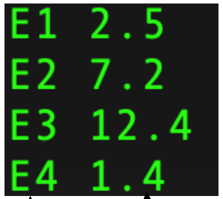
cwl-demo

- How do you calculate the expected cost, & expected total cost?
  - To calculate costs you will need to do two things.
    - Denote which result item is of what result type in the result_file



```
T1   E2   D1   1    4.3 R1
T1   E2   D2   2    4.2 R1
T1   E1   D3   3    4.1 R1
```

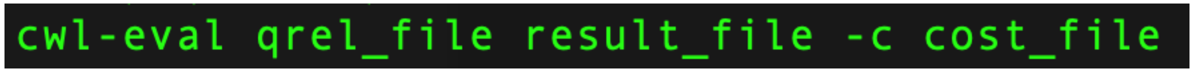
    - Include a cost file that maps result type to a cost in a cost_file



```
E1 2.5
E2 7.2
E3 12.4
E4 1.4
```

cwl-demo

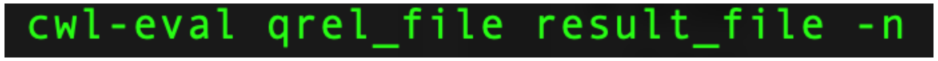
- How do you calculate the expected cost, & expected total cost?
  - Once you have the mappings you can add **”-c cost_file”** to then calculate the costs.
  - Note that you determine the units of the costs
  - Where you could use:
    - Time (sec)
    - Length (number of chars, or number of terms)
    - Dollars and cents



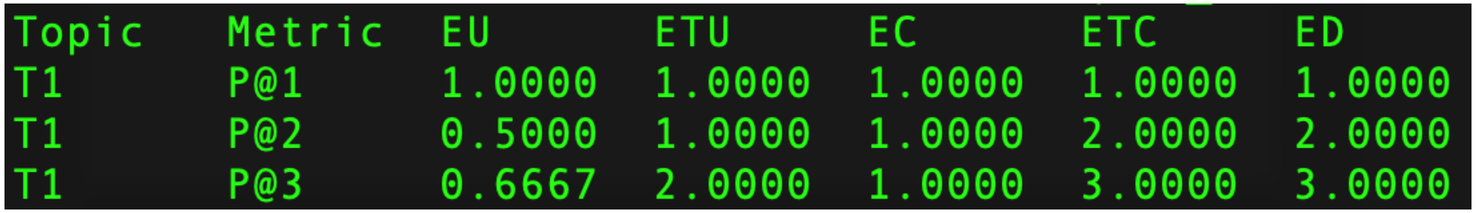
```
cwl-eval qrel_file result_file -c cost_file
```

cwl-demo

- How to show the column names?
  - If you'd like to show the names for each column include **"-n"**

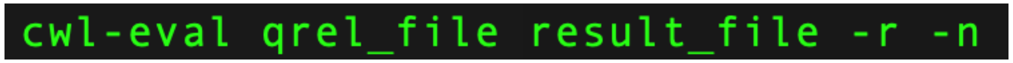


```
cwl-eval qrel_file result_file -n
```

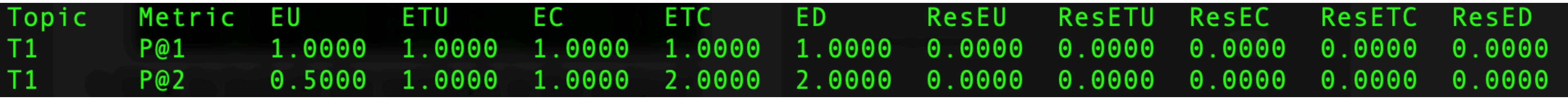


```
Topic    Metric   EU       ETU      EC       ETC      ED
T1       P@1      1.0000   1.0000   1.0000   1.0000   1.0000
T1       P@2      0.5000   1.0000   1.0000   2.0000   2.0000
T1       P@3      0.6667   2.0000   1.0000   3.0000   3.0000
```

- How to include the residuals in the output?
  - If you'd like to show the residuals include **"-r"**



```
cwl-eval qrel_file result_file -r -n
```



```
Topic   Metric  EU      ETU     EC      ETC     ED      ResEU   ResETU  ResEC   ResETC  ResED
T1      P@1     1.0000  1.0000  1.0000  1.0000  1.0000  0.0000  0.0000  0.0000  0.0000  0.0000
T1      P@2     0.5000  1.0000  1.0000  2.0000  2.0000  0.0000  0.0000  0.0000  0.0000  0.0000
```

cwl-demo

- How specify the metrics you want?
  - If you want to control what measures you want to output then specify them **“-m file.to.metrics”**

```
NDCGCWLMetric(10)
RBPCWLMetric(0.25)
INSTCWLMetric(2)
```

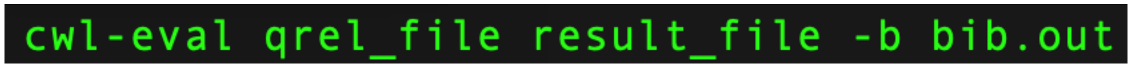
CWL Metrics with param values

```
cwl-eval qrel_file result_file -m mixed.metrics
```

```
T1      NDCG-k@10       0.5645  2.5650  1.0000  4.5436  4.5436
T1      RBP@0.25        0.8088  1.0784  1.0000  1.3333  1.3333
T1      INST-T=2        0.5994  1.7079  1.0000  2.8475  2.8496
```

cwl-eval demo

- How to get the BibTeX for the metrics?
  - If you want cwl-eval to provide you with the list of references to the metrics that you have specified in your metrics file then you can use the "–b" flag and specify where you want the BibTeX to be saved to.



```
cwl-eval qrel_file result_file -b bib.out
```

- How to cite cwl eval ☺ ?
  - Azzopardi, Thomas and Moffat, cwl_eval: An Evaluation Tool for Information Retrieval, Proc. of the 42nd International ACM SIGIR Conference, SIGIR2019

cwl-eval demo

- What are the differences between **cwl-eval** and **trec-eval**?
  - trec_eval re-orders the results based on the score.
  - trec_eval is not gain sensitive (binary relevance).
  - In cwl-eval a relevance value of 2, is interpreted as 2 units of gain
  - cwl-eval reports all measures in the same units – you can directly compare the estimates by each metric.
  - cwl-eval reports EU, ETU, EC, ETC, ED and can also report residuals outputting them into a Pandas dataframe friendly format.

User Models, Metrics and Measures of Search: A Tutorial on the CWL Evaluation Framework ACM CHIIR UMMMS 2021

by
Leif Azzopardi, Alistair Moffat, Paul Thomas and Guido Zuccon

https://github.com/ireval/cwl

ADDING YOUR OWN CWL METRIC

cwl classes

- Ranking
  - Is a class to represent the costs and gains for a query
- cwl_ruler/CWLRuler
  - Is a container class that creates all the CWLMetrics
- measures/cwl_metrics/CWLMetrics
  - Base class for creating new metrics
  - Subclass, and then define the
    - c_vector()
    - Or define the w_vector() and convert the w's to c's
  - Define the metric_name for reporting

DCG - Example

```
def name(self):
    return "NDCG-k@{0}".format(self.k)

def c_vector(self, ranking, worse_case=True):

    cvec = []
    for i in range(1, ranking.n+1):
        if i < self.k:
            cvec.append(math.log(i+1, self.base)/math.log(i+2, self.base))
        else:
            cvec.append(0.0)

    cvec = np.array(cvec)

    return cvec
```

Recall that C_DCG = $\log_b(i+1)/\log_b(i+2)$

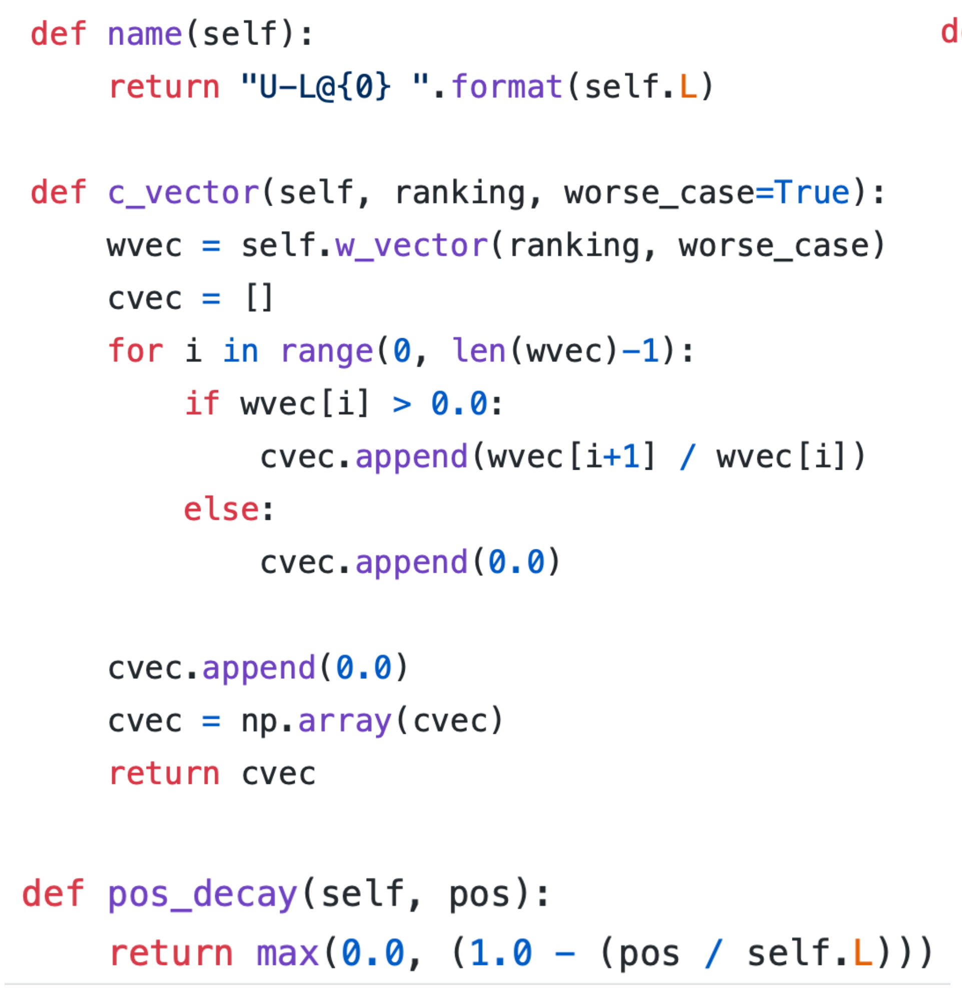
RBP - Example

```
def name(self):
    return "RBP@{0}".format(self.theta)

def c_vector(self, ranking, worse_case=True):
    cvec = np.dot(np.ones(ranking.n), self.theta)
    return cvec
```

Recall that C_RBP = theta

U-Measure Example

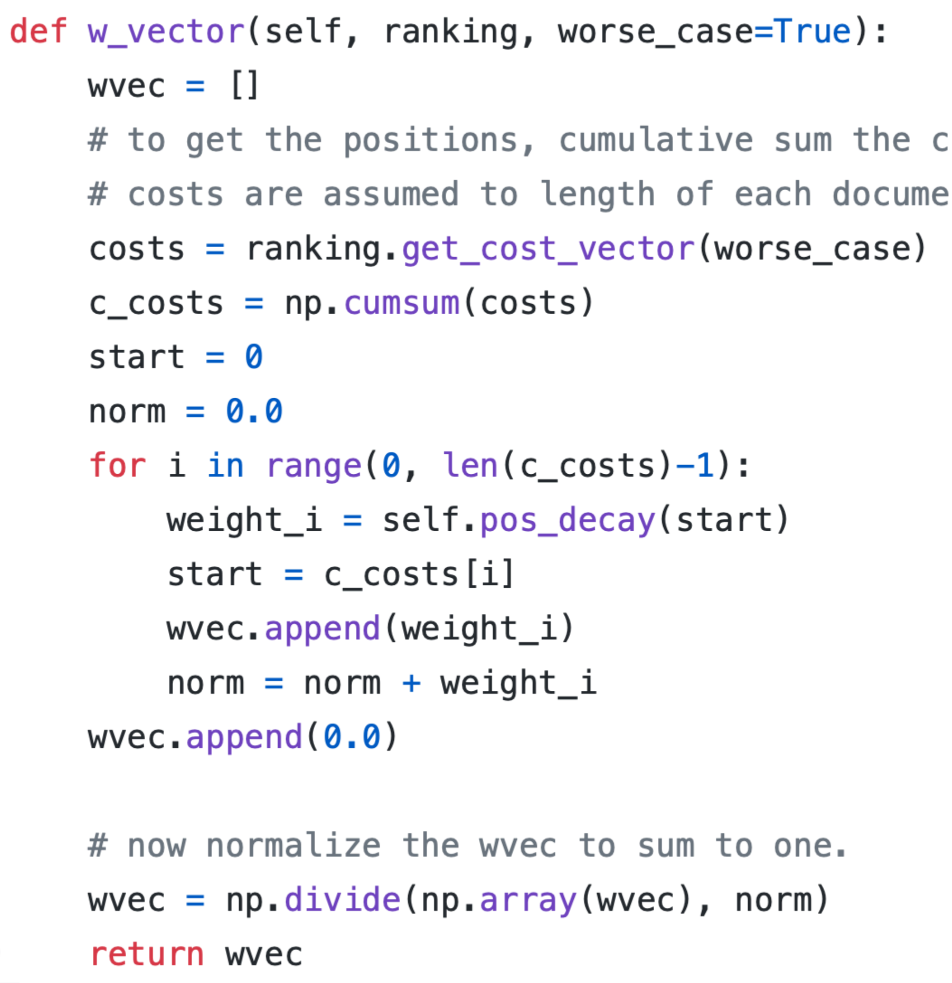


```
def name(self):
    return "U-L@{0} ".format(self.L)

def c_vector(self, ranking, worse_case=True):
    wvec = self.w_vector(ranking, worse_case)
    cvec = []
    for i in range(0, len(wvec)-1):
        if wvec[i] > 0.0:
            cvec.append(wvec[i+1] / wvec[i])
        else:
            cvec.append(0.0)

    cvec.append(0.0)
    cvec = np.array(cvec)
    return cvec

def pos_decay(self, pos):
    return max(0.0, (1.0 - (pos / self.L)))
```



```
def w_vector(self, ranking, worse_case=True):
    wvec = []
    # to get the positions, cumulative sum the c
    # costs are assumed to length of each docume
    costs = ranking.get_cost_vector(worse_case)
    c_costs = np.cumsum(costs)
    start = 0
    norm = 0.0
    for i in range(0, len(c_costs)-1):
        weight_i = self.pos_decay(start)
        start = c_costs[i]
        wvec.append(weight_i)
        norm = norm + weight_i
    wvec.append(0.0)

    # now normalize the wvec to sum to one.
    wvec = np.divide(np.array(wvec), norm)
    return wvec
```

For the U-Measure I don’t know what the C vector is.
But I can define the W and transform it to the C

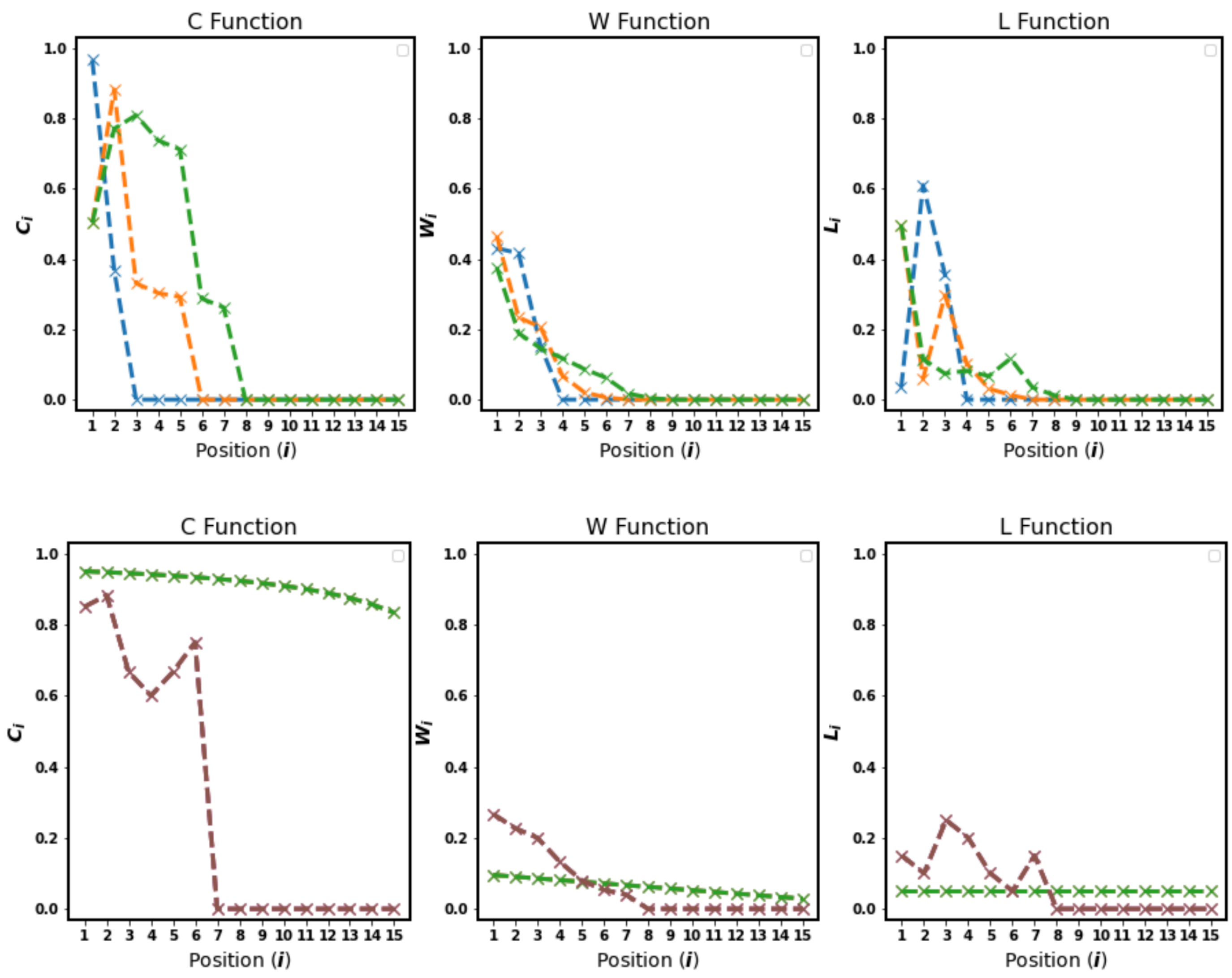
User Models, Metrics and Measures of Search: A Tutorial on the CWL Evaluation Framework ACM CHIIR UMMMS 2021

by
Leif Azzopardi, Alistair Moffat, Paul Thomas and Guido Zuccon

http://bit.ly/cwl-chiir-2021-notebook-demo

INSIDE CWL METRICS

Check out the notebook to see CWL graphs like this!!!


C Function
W Function
L Function
C_i
W_i
L_i
Position (i)