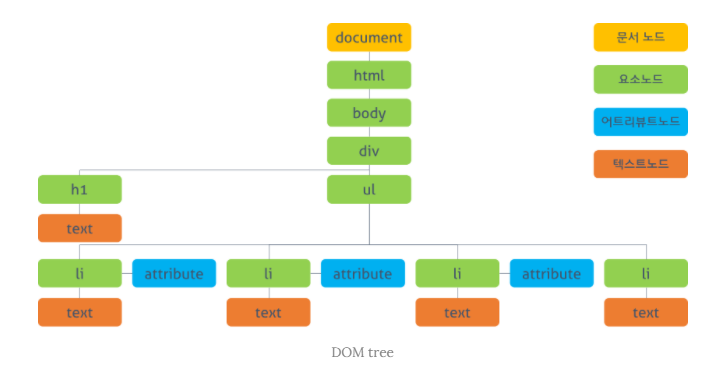
1. Content Extraction via Text Density(CETD), 2011년 논문  
   🡺 extract the main content + preserve its original structure information

■ DOM tree 구조

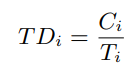
■ 2 measures

① Text Density  
- HTML 문서가 parsed되고 DOM tree에 의해 표시되면, 각 노드의 text density를 계산한다.  
- 높은 text density : 그 노드가 웹 페이지 내에서 content-text가 포함된 tag일 가능성이 높음  
- 낮은 text density : noise

② Composite Text Density  
- 하이퍼링크에 대한 통계적인 정보를 추가 : Text Density를 Composite Text Density로 확장  
- content에서 낮은 text density node 손실 문제를 해결하기 위해, DensitySum 기술이 integral content를 추출하기 위해 고안되었다.  
🡺 빠르고, 정확한 결과 (성능이 일반 content 추출 알고리즘보다 우수)  
🡺 웹 페이지의 특정 HTML 구조를 찾지 않는다.

■ 3.2 Text Density 아키텍처

1. HTML 문서가 parsed되고 표시  
🡺 각각의 노드가 나타내는 글자 수 와 tag 수 파악 가능  
🡺 통계적 접근 (CharNumber(subtree의 모든 글자 수), TagNumber(subtree의 모든 tag수))

2. character와 tag 의 비율 계산

🡺 Ti 가 0이면 1로 설정

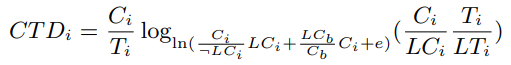
- 보통 long, simply formatted text 🡺 높은 값 (상대적으로 의미 있는 content)  
- highly formatted nodes, brief text 🡺 낮은 값

□ 계산하기 전에, script, comment, style 태그는 제거.  
□ 특정 노드가 태그수가 0이면, 밀도는 노드가 포함하는 문자수로 설정  
□ Compute-Density 알고리즘은 Algorithm 1로 묘사, N은 계산중인 DOM 노드

■ 3.3 Composite Text Density

- 대부분의 noise는 하이퍼링크를 포함  
🡺 통계적 접근   
1. LinkCharNumber(subtree의 모든 하이퍼링크 글자 수)  
2. LinkTagNumber(subtree의 모든 하이퍼링크 tag수))

- Text Density 재정의



LCi : 모든 하이퍼링크 글자수  
ㄱLCi : non-hyperlink 글자수  
LTi : 하이퍼링크 tag수  
LCb : <body> tag에서의 하이퍼링크 글자 수  
Cb : <body> tag에서의 모든 글자수

 : 하이퍼링크 text의 비율

: 하이퍼링크 tag의 비율

- high value : 태그 i가 수많은 non-hyperlink character와 hyperlink가 거의 없다

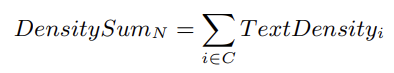
■ 4. Content Extraction

노드를 content 또는 noise로 나누는 Threshold t 결정 🡺 extract the content  
( text density가 t보다 크거나 같으면, content로 labled받는다.)  
🡺 t값 결정 (content만 완전히 추출하는) HOW ?

- <body> 태그(우리가 계산하는root node)   
noise보다 content가 더 많이 포함 & 의미 있는 content보다 hyplerlink 더 많이 포함  
🡺 text density는 둘을 구별하기에 충분한 중간값  
🡺 <body> 태그의 text density를 threshold로 설정하였음

■ 4.2 DensitySum

- picture, hyperlink, 뉴스기사의 byline 또는 dateline 등 🡺 낮은 text density (비정상적)  
 역으로, 저작권/ 각서정보 등 비정상적으로 높은 text density  
- 따라서, 단순히 threshold 따라 노드를 라벨링 하면, content node가 손실 & 일부 noise 유지  
🡺 HOW?

- DensitySum : data smoothing – outlying peak 와 valley를 평활하게 하고 section 내, section 간의 차이를 증가시켜 문제 해결 가능  
content block(DOM의 구조의 상위 노드)는 하위 노드의 text density를 추가하면 peak value를 얻는다.  
 N이 웹 페이지의 tag  
i가 N의 하위

🡺 maximum DensitySum으로 <body> 태그를 지정 후, content로 표시  
🡺 그 후에 its ancestor(조상)과 subtree(하위) 트리를 유지함으로써 content를 추출 가능

□ 대부분 웹 페이지에 2개 이상의 content block 존재  
🡺 따라서, text density가 threshold보다 큰 노드에 대해 위의 방법 적용하여 내용 추출

□ 일부 웹 페이지 content text density < body text density  
🡺 단순히 body text density를 threshold로 사용하면 손실됨  
🡺 해결

1. 전체 페이지에 Maximum DensitySum tag 를 찾는다. (threshold 없이)  
2. Maximum DensitySum tag 로부터 <body> tag로의 경로에 있는 노드의 minimum text density를 설정  
🡪 Algorithm2 참고

■ 정리

3가지 알고리즘 구현

1. Text density with DensitySum method (CETD-DS)
2. Composite Text density with Data Smoothing method (CETD-S)  
   (형제 노드를 가중 평균화)
3. Composite Text Density with DensitySum method (CECTD-DS)

참고 문헌   
[DOM Based Content Extraction via Text Density]  
<http://www.ofey.me/papers/cetd-sigir11.pdf> ,