Overview of the TREC 2022 Fair Ranking Track

Michael D. Ekstrand
michaelekstrand@boisestate.edu

Graham McDonald
graham.mcdonald@glasgow.ac.uk

Amifa Raj
amifaraj@u.boisestate.edu

Isaac Johnson
isaac@wikimedia.org

March 22, 2023

Contents

1 Introduction

The TREC Fair Ranking Track aims to provide a platform for participants to develop and evaluate novel retrieval algorithms that can provide a fair exposure to a mixture of demographics or attributes, such as ethnicity, that are represented by relevant documents in response to a search query. For example, particular demographics or attributes can be represented by the documents' topical content or authors.

The 2022 Fair Ranking Track adopted a resource allocation task. The task focused on supporting Wikipedia editors who are looking to improve the encyclopedia's coverage of topics under the purview of a WikiProject.[1] WikiProject coordinators and/or Wikipedia editors search for Wikipedia documents that are in need of editing to improve the quality of the article. The 2022 Fair Ranking track aimed to ensure that documents that are about, or somehow represent, certain protected characteristics receive a fair exposure to the Wikipedia editors, so that the documents have an fair opportunity of being improved and, therefore, be well-represented in Wikipedia. The under-representation of particular protected characteristics in Wikipedia can result in systematic biases that can have a negative human, social, and economic impact, particularly for disadvantaged or protected societal groups [4, 7].

[1] https://en.wikipedia.org/wiki/WikiProject

2 Task Definition

The 2022 Fair Ranking Track used an *ad hoc* retrieval protocol. Participants were provided with a corpus of documents (a subset of the English language Wikipedia) and a set of queries. A query was of the form of a short list of search terms that represent a WikiProject. Each document in the corpus was relevant to zero to many WikiProjects and associated with zero to many fairness categories.

There were two tasks in the 2022 Fair Ranking Track. In each of the tasks, for a given query, participants were to produce document rankings that are:

1. Relevant to a particular WikiProject.
2. Provide a fair exposure to articles that are associated to particular protected attributes.

The tasks shared a topic set, the corpus, the basic problem structure and the fairness objective. However, they differed in their target user persona, system output (static ranking vs. sequences of rankings) and evaluation metrics. The common problem setup was as follows:

- **Queries** were provided by the organizers and derived from the topics of existing or hypothetical WikiProjects.
- **Documents** were Wikipedia articles that may or may not be relevant to any particular WikiProject that is represented by a query.
- **Rankings** were ranked lists of articles for editors to consider working on.
- **Fairness** of exposure should be achieved with respect to the protected attributes associated with the documents. Documents can be associated to many different fairness attributes. The official track evaluation focused on intersectional fairness and, as such, evaluated how fairly systems rank documents with respect to all of the fairness categories. However, individual teams could choose whether to optimise their systems with respect to all, a subset of, or individual fairness categories.

2.1 Task 1: WikiProject Coordinators

The first task focused on WikiProject coordinators as users of the search system; their goal is to search for relevant articles and produce a ranked list of articles needing work that other editors can then consult when looking for work to do.

Output: The output for this task was a **single ranking per query**, consisting of **500 articles**.

Evaluation was a multi-objective assessment of rankings by the following two criteria:

- Relevance to a WikiProject topic. We will provide relevance assessments for the articles derived from existing Wikipedia data; Ranking relevance will be computed with nDCG, using binary relevance and logarithmic decay.
- Fairness with respect to the exposure of different fairness categories associated to the articles returned in response to a query.

Section 4.2 contains details on the evaluation metrics.

2.2 Task 2: Wikipedia Editors

The second task focused on individual Wikipedia editors looking for work associated with a project. The conceptual model is that rather than maintaining a fixed work list as in Task 1, a WikiProject coordinator would create a saved search, and when an editor looks for work they re-run the search. This means that different editors may receive different rankings for the same query, and differences in these rankings may be leveraged for providing fairness.

Output: The output of this task is **100 rankings per query**, each consisting of **20 articles**.

Evaluation was a multi-objective assessment of rankings by the following three criteria:

- Relevance to a WikiProject topic. We will provide relevance assessments for articles derived from existing Wikipedia data. Ranking relevance will be computed with nDCG.
- Work needed on the article (articles needing more work preferred). We provide the output of an article quality assessment tool for each article in the corpus; for the purposes of this track, we assume lower-quality articles need more work.
- Fairness with respect to the exposure of different fairness categories associated to the articles returned in response to a query.

The goal of this task was *not* to be fair to work-needed levels; rather, we consider work-needed and topical relevance to be two components of a multi-objective notion of relevance, so that between two documents with the same topical relevance, the one with more work needed is more relevant to the query in the context of looking for articles to improve.

This task used *expected exposure* to compare the exposure article subjects receive in result rankings to the *ideal* (or *target*) *exposure* they would receive based on their relevance and work-needed [1]. This addresses fundamental limits in the ability to provide fair exposure in a single ranking by examining the exposure over multiple rankings.

For each query, participants provided 100 rankings, which we considered to be samples from the distribution realized by a stochastic ranking policy (given a query q, a distribution π_q over truncated permutations of the documents). Note that this is how we interpret the queries, but it did not mean that a stochastic policy is how the system should have been implemented — other implementation designs were certainly possible. The objective was to provide equitable exposure to documents of comparable relevance and work-needed, aggregated by protected attribute. Section 4.3 has details on the evaluation metrics.

3 Data

This section provides details of the format of the test collection, topics and ground truth. Further details about data generation and limitations can be found in Section 5.2.

3.1 Obtaining the Data

The corpus and query data set is distributed via Globus, and can be obtained in two ways. First, it can be obtained via Globus, from our repository at `https://boi.st/TREC2022Globus`. From this site, you can log in using your institution's Globus account or your own Google account, and synchronize it to your local Globus install or download it with Globus Connect Personal.[2] This method has robust support for restarting downloads and dealing with intermittent connections. Second, it can be downloaded directly via HTTP from: `https://data.boisestate.edu/library/Ekstrand/TRECFairRanking/2022/`.

The runs and evaluation qrels will be made available in the ordinary TREC archives.

[2] `https://www.globus.org/globus-connect-personal`

3.2 Corpus

The corpus consisted of articles from English Wikipedia. We removed all redirect articles, but left the wikitext (markup Wikipedia uses to describe formatting) intact. This was provided as a JSON file, with one record per line, and compressed with gzip (`trec_corpus.json.gz`). Each record contains the following fields:

id The unique numeric Wikipedia article identifier.

title The article title.

url The article URL, to comply with Wikipedia licensing attribution requirements.

The three available formats of the corpus are as follows:

text: The full article text, with Wiki markup (`text` file only)

plain: The full article text, without Wiki markup (`plain` file only)

html: The full article text, rendered into HTML (`html` file only)

The contents of this corpus were prepared in accordance with, and licensed under, the CC BY-SA 3.0 license.[3] The raw Wikipedia dump files used to produce this corpus are available in the `source` directory; this is primarily for archival purposes, because Wikipedia does not publish dumps indefinitely.

3.3 Queries

The queries are in the `2022` directory, in the file `train_topics_meta.jsonl`. Each of the queries map to a single Wikiproject. The primary query texts are constructed from extracted keywords from articles that are relevant to a Wikiproject. The following fields are provided:

id A query identifier (int)

title The Wikiproject title (string)

keywords A collection of search keywords forming the query text (list of str). We cleaned and parsed the Wiki articles and then used KeyBert [2] to extract the most representative words of those articles. For each Wikiproject, we aggregated the extracted keywords from relevant articles and, after some manual filtering, used those as query texts for that particular Wikiproject.

url The URL for the Wikiproject. This is provided for attribution and not expected to be used by your system as it will not be present in the evaluation data (string)

rel_docs A list of the page IDs of relevant pages (list of int)

The eval-queries are also in the `2022` directory, in the file `eval_topics.jsonl` which has *id*, *keywords*, and *title* fields.

In addition to query relevance, for Task 2: Wikipedia Editors (Section 2.2), participants will also be expected to return relevant documents that need more editing work done more highly than relevant documents that need less work done.

[3] `https://creativecommons.org/licenses/by-sa/3.0/`

3.4 Fairness Categories

Fairness ground truth labels for the following fairness categories are also in the 2022 directory, in the `trec_2022_articles_discrete.json.gz` file. While we provide the raw values for each fairness category with the data, for most categories we also map the raw values to a reduced, fixed set of categories that will be used to judging systems.[4]

Geographic location (article topic) The geographical location associated with the article topic. Both the associated countries—e.g., United Kingdom—and sub-continental regions—e.g., Northern Europe—are provided but systems will be evaluated using sub-continental regions (and not countries). An article can have 0 to many regions associated with it.

Geographic location (article sources) The geographic location associated with the article based on the article's sources. Same categories as article geographic location above.

Gender (biographies only) The gender of the individual about which the biography pertains. Gender has been reduced to four distinct categories: Man, Woman, Non-binary, and Unknown (missing data or not a biography).

Age of the topic How old the subject of the article is. For example, the birth date of a person in a biographical article, the date that an event occurred for articles that are about an event, or the creation date of a piece of art or music when the article is about the piece of art or music. The raw years are mapped to four distinct categories: Unknown, Pre-1900s, 20th century, and 21st century.

Occupation (biographies only) The occupation of the subject of an article. An article have 0 (unknown) to many occupations associated with it. There are 32 distinct occupation categories included in the data.

Alphabetical Editors often work through articles in alphabetical order and this can result in articles about subjects / topics that start with letters that appear earlier in the alphabet getting more exposure to the editors. Therefore, it is important that articles from later in the alphabet also get a fair exposure to the editors. The first letter is mapped to four discrete categories: a-d, e-k, l-r, and s-.

Age of the article The length of time the article has existed. The date is mapped to one of four discrete categories: 2001-2006, 2007-2011, 2012-2016, and 2017-2022.

Popularity (# pageviews) Number of times the page was viewed in February 2022. The number of pageviews are normalized and mapped to four discrete categories: Low, Medium-Low, Medium-High, and High.

Replication of articles in other languages The number of other language Wikipedias that the article is replicated in. This can range from English-only to all 300+ languages of Wikipedia but is mapped to three discrete categories: English only, 2-4 languages, and 5+ languages.

For the purposes of multidimensional fairness, we treated the dimensions as independent, and took the outer product of the fairness categories as a combined fairness space. The resulting space had over 11M dimensions, requiring care in implementing target alignments and metrics.

3.5 Metadata

We provide a simple Wikimedia quality score (a float between 0 and 1 where 0 is no content on the page and 1 is high quality) for optimizing for work-needed in Task 2. Work-needed can be operationalized as the reverse—i.e. 1 minus this quality score. The discretized quality scores will be used as work-needed for final system evaluation.

[4]For more information, see: `https://public.paws.wmcloud.org/User:Isaac_(WMF)/TREC/TREC_2022_Data.ipynb`

This data is provided together in a metadata file (`trec_metadata.json.gz`), in which each line is the metadata for one article represented as a JSON record with the following keys:

page_id Unique page identifier (int)

quality_score Continuous measure of article quality with 0 representing low quality and 1 representing high quality (float in range $[0, 1]$)

quality_score_disc Discrete quality score in which the quality score is mapped to six ordinal categories from low to high: Stub, Start, C, B, GA, FA (string)

Group Alignments The group alignments associated to an article as described in Section 3.4.

3.6 Output

For **Task 1**, participants outputted results in rank order in a tab-separated file with two columns:

id The query ID for the topic

page_id ID for the recommended article

For **Task 2**, this file had 3 columns, to account for repeated rankings per query:

id Query ID

rep_number Repeat Number (1-100)

page_id ID for the recommended article

4 Evaluation Metrics

Each task was evaluated with its own metric designed for that task setting. The goal of these metrics was to measure the extent to which a system (1) exposed relevant documents, and (2) exposed those documents in a way that is fair to article topic groups, defined by the mentioned fairness constraints of the article's subject.

This faces a problem in that Wikipedia itself has well-documented biases: if we target the current group distribution within Wikipedia, we will reward systems that simply reproduce Wikipedia's existing biases instead of promoting social equity. However, if we simply target equal exposure for groups, we would ignore potential real disparities in topical relevance. Due to the biases in Wikipedia's coverage, and the inability to retrieve documents that don't exist to fill in coverage gaps, there is not good empirical data on what the distribution for any particular topic *should* be if systemic biases did not exist in either Wikipedia or society (the "world as it could and should be" [3]). Therefore, in this track we adopted a compromise: we **averaged** the empirical distribution of groups among relevant documents with the world population (for location) or equality (for gender) to derive the target group distribution.

Code to implement the metrics is found at `https://github.com/fair-trec/trec2022-fair-public`.

4.1 Preliminaries

The tasks were to retrieve documents d from a corpus $\mathcal{D}$ that are relevant to a query q. $\mathbf{r}_q \in [0, 1]^{|\mathcal{D}|}$ is a vector of relevance judgements for query q. We denote a ranked list by L; L_i is the document at position i (starting from 1), and L_d^{-1} is the rank of document d. For Task 1, each system returned a single ranked list; for Task 2, it returned a sequence of rankings $\mathcal{L}$.

We represented the group alignment of a document d with an *alignment vector* $\mathbf{a}_d \in [0,1]^{|\mathcal{G}|}$. a_{dg} is document d's alignment with group g. $\mathbf{A} \in [0,1]^{|\mathcal{D}|\times|\mathcal{G}|}$ is the alignment matrix for all documents. $\mathbf{a}_{\text{world}}$ denotes the distribution of the world.[5]

We considered fairness with respect to two group sets, $\mathcal{G}_{\text{geo}}$ and $\mathcal{G}_{\text{gender}}$. We operationalized this intersectional objective by letting $\mathcal{G} = \mathcal{G}_{\text{geo}} \times \mathcal{G}_{\text{gender}}$, the Cartesian product of the two group sets. Further, alignment under either group set may be unknown; we represented this case by treating "unknown" as its own group ($g_?$) in each set. In the product set, a document's alignment may be unknown for either or both groups.

In all metrics, we use **log discounting** to compute attention weights:

$$v_i = \frac{1}{\log_2 \max(i, 2)}$$

Task 2 also considered the work each document needs, represented by $w_d \in \{1, 2, 3, 4\}$.

4.2 Task 1: WikiProject Coordinators (Single Rankings)

For the single-ranking Task 1, we adopted attention-weighted rank fairness (AWRF), first described by Sapiezynski et al. [8] and named by Raj et al. [6]. AWRF computes a vector $\mathbf{d}_L$ of the cumulated exposure a list gives to each group, and a target vector $\mathbf{d}^*_q$; we then compared these with the Jenson-Shannon divergence:

$$\begin{aligned}
\mathbf{d}'_L &= \sum_i v_i \mathbf{a}_{L_i} && \text{cumulated attention} \\
\mathbf{d}_L &= \frac{\mathbf{d}'_L}{\|\mathbf{d}'_L\|_1} && \text{normalize to a distribution} \\
\mathbf{d}^*_q &= \frac{1}{2}\left(\mathbf{A}^{\mathrm{T}}\mathbf{r}_q + \mathbf{a}_{\text{world}}\right) \\
\text{AWRF}(L) &= 1 - \mathsf{d}_{\text{JS}}(\mathbf{d}_L, \mathbf{d}^*_q) && (1)
\end{aligned}$$

For Task 1, we ignored documents that are fully unknown for the purposes of computing $\mathbf{d}_L$ and $\mathbf{d}^*_q$; they do not contribute exposure to any group.

The resulting metric is in the range $[0, 1]$, with 1 representing a maximally-fair ranking (the distance from the target distribution is minimized). We combined it with an ordinary nDCG metric for utility:

$$\text{NDCG}(L) = \frac{\sum_i v_i r_{qd}}{\text{ideal}} \tag{2}$$

$$M_1(L) = \text{AWRF}(L) \times \text{NDCG}(L) \tag{3}$$

To score well on the final metric M_1, a run must be **both** accurate and fair.

4.3 Task 2: Wikipedia Editors (Multiple Rankings)

For Task 2, we used Expected Exposure [1] to compare the exposure each group receives in the sequence of rankings to the exposure it would receive in a sequence of rankings drawn from an *ideal policy* with the following properties:

- Relevant documents come before irrelevant documents
- Relevant documents are sorted in nonincreasing order of work needed

[5]Obtained from `https://en.wikipedia.org/wiki/List_of_continents_and_continental_subregions_by_population`

- Within each work-needed bin of relevant documents, group exposure is fairly distributed according to the average of the distribution of relevant documents and the distribution of global population (the same average target as before).

We have encountered some confusion about whether this task is requiring fairness towards work-needed; as we have designed the metric, work-needed is considered to be a part of (graded) relevance: a document is more relevant if it is relevant to the topic and needs significant work. In the Expected Exposure framework, this combined relevance is used to derive the target policies.

To apply expected exposure, we first define the exposure ϵ_d a document d receives in sequence $\mathcal{L}$:

$$\epsilon_d = \frac{1}{|\mathcal{L}|} \sum_{L \in \mathcal{L}} w_{L_d^{-1}} \tag{4}$$

This forms an exposure vector $\boldsymbol{\epsilon} \in \mathbb{R}^{|\mathcal{D}|}$. It is aggregated into a group exposure vector $\boldsymbol{\gamma}$, including "unknown" as a group:

$$\boldsymbol{\gamma} = \mathbf{A}^{\mathrm{T}} \boldsymbol{\epsilon} \tag{5}$$

Our implementation rearranges the mean and aggregate operations, but the result is mathematically equivalent.

We then compare these system exposures with the target exposures $\boldsymbol{\epsilon}^*$ for each query. This starts with the per-document ideal exposure; if m_w is the number of relevant documents with work-needed level $w \in \{1, 2, 3, 4\}$, then according to Diaz et al. [1] the ideal exposure for document d is computed as:

$$\epsilon_d^* = \frac{1}{m_{w_d}} \sum_{i=m_{>w_d}+1}^{m_{\geq w_d}} v_i \tag{6}$$

We use this to compute the non-averaged target distribution $\tilde{\boldsymbol{\gamma}}^*$:

$$\tilde{\boldsymbol{\gamma}}^* = \mathbf{A}^{\mathrm{T}} \boldsymbol{\epsilon}^* \tag{7}$$

Since we include "unknown" as a group, we have a challenge with computing the target distribution by averaging the empirical distribution of relevant documents and the global population — global population does not provide any information on the proportion of relevant articles for which the fairness attributes are relevant. Our solution, therefore, is to average the distribution of *known-group* documents with the world population, and re-normalize so the final distribution is a probability distribution, but derive the proportion of known- to unknown-group documents entirely from the empirical distribution of relevant documents. Extended to handle partially-unknown documents, this procedure proceeds as follows:

- Average the distribution of fully-known documents (both gender and location are known) with the global intersectional population (global population by location and equality by gender).
- Average the distribution of documents with unknown location but known gender with the equality gender distribution.
- Average the distribution of documents with unknown gender but known location with the world population.

The result is the target group exposure $\boldsymbol{\gamma}^*$. We use this to measure the **expected exposure loss**:

$$\begin{aligned} M_2(\mathcal{L}_q) &= \|\boldsymbol{\gamma} - \boldsymbol{\gamma}^*\|_2 \qquad (8) \\ &= \boldsymbol{\gamma} \cdot \boldsymbol{\gamma} - 2\boldsymbol{\gamma} \cdot \boldsymbol{\gamma}^* + \boldsymbol{\gamma}^* \cdot \boldsymbol{\gamma}^* \\ \text{EE-D}(\mathcal{L}_q) &= \boldsymbol{\gamma}^* \cdot \boldsymbol{\gamma}^* \qquad (9) \\ \text{EE-R}(\mathcal{L}_q) &= \boldsymbol{\gamma} \cdot \boldsymbol{\gamma}^* \qquad (10) \end{aligned}$$

	nDCG	AWRF	Score	95% CI
tmt5	0.7242	0.4988	0.3626	(0.326, 0.397)
UoGRelvOnlyT1	0.6044	0.5246	0.3254	(0.284, 0.372)
UoGTrT1ColPRF	0.6044	0.5246	0.3254	(0.283, 0.369)
UoGTrExpE2	0.5977	0.5243	0.3230	(0.280, 0.368)
0mt5	0.6216	0.4778	0.2990	(0.267, 0.332)
0mt5_p	0.5841	0.5015	0.2949	(0.262, 0.326)
tmt5_p	0.5728	0.5121	0.2946	(0.260, 0.327)
FRT_constraint	0.5749	0.4793	0.2782	(0.245, 0.312)
bm25_p	0.5434	0.5026	0.2773	(0.241, 0.312)
UoGTrQE	0.5368	0.4983	0.2734	(0.240, 0.309)
UoGTrExpE1	0.5176	0.5122	0.2716	(0.238, 0.308)
UDInfo_F_bm25	0.5666	0.4719	0.2708	(0.236, 0.302)
ans_bm25	0.5661	0.4719	0.2706	(0.237, 0.303)
UDInfo_F_mlp2	0.5655	0.4718	0.2703	(0.235, 0.302)
FRT_attention	0.5893	0.4484	0.2702	(0.231, 0.311)
UDInfo_F_lgbm2	0.5645	0.4719	0.2698	(0.235, 0.302)
UDInfo_F_mlp4	0.5638	0.4719	0.2695	(0.234, 0.301)
UDInfo_F_lgbm4	0.5631	0.4723	0.2693	(0.235, 0.302)
FRT_diversity	0.5305	0.4909	0.2641	(0.229, 0.299)
rmit_cidda_ir_5	0.5417	0.4525	0.2485	(0.215, 0.282)
rmit_cidda_ir_1	0.5438	0.4416	0.2433	(0.210, 0.277)
rmit_cidda_ir_4	0.5388	0.4435	0.2431	(0.209, 0.278)
rmit_cidda_ir_7	0.5382	0.4443	0.2426	(0.209, 0.276)
rmit_cidda_ir_3	0.5365	0.4447	0.2420	(0.208, 0.275)
rmit_cidda_ir_6	0.5343	0.4457	0.2418	(0.208, 0.276)
rmit_cidda_ir_8	0.5322	0.4469	0.2415	(0.208, 0.276)
rmit_cidda_ir_2	0.5197	0.4443	0.2345	(0.201, 0.269)

Table 1: Task 1 runs. Higher score is better (for all metrics). CI is 95% bootstrapped CI of score.

Lower M_2 is better. It decomposes into two submetrics, the **expected exposure disparity** (EE-D) that measures overall inequality in exposure independent of relevance, for which lower is better; and the **expected exposure relevance** (EE-L) that measures exposure/relevance alignment, for which higher is better [1].

5 Results

This year 5 different teams submitted a total of 24 runs. All 5 teams participated in Task 1: Single Rankings (27 runs total), while 2 groups participated in Task 2: Multiple Rankings (11 runs total).

5.1 Task 1: WikiProject Coordinators (Single Rankings)

Approaches for Task 1 included:

- Relevance ranking by ColBERT-PRF.
- Relevance ranking by ColBERT-E2E and a heuristic approach that re-ranks to match target exposure using diversification.
- Query rewriting strategy to expand query.

	Overall	age	alpha	gender	langs	occ	pop	src-geo	sub-geo
tmt5	**0.3626**	**0.6860**	**0.7190**	**0.6795**	**0.6786**	**0.6825**	**0.6313**	**0.6453**	**0.6450**
UoGRelvOnlyT1	0.3254	0.5843	0.5916	0.5266	0.5896	0.5267	0.5802	0.5548	0.5389
UoGTrT1ColPRF	0.3254	0.5843	0.5916	0.5266	0.5896	0.5267	0.5802	0.5548	0.5389
UoGTrExpE2	0.3230	0.5797	0.5869	0.5453	0.5849	0.5443	0.5765	0.5482	0.5344
0mt5	0.2990	0.5833	0.6164	0.5834	0.5817	0.5845	0.5333	0.5529	0.5496
0mt5_p	0.2949	0.5552	0.5801	0.5519	0.5530	0.5514	0.5112	0.5315	0.5352
tmt5_p	0.2946	0.5481	0.5696	0.5437	0.5463	0.5421	0.5077	0.5271	0.5303
FRT_constraint	0.2782	0.5511	0.5712	0.5330	0.5494	0.5358	0.5220	0.5144	0.5059
bm25_p	0.2773	0.5178	0.5395	0.5156	0.5165	0.5147	0.4768	0.4969	0.5005
UoGTrQE	0.2734	0.5216	0.5291	0.4813	0.5225	0.4807	0.4975	0.4828	0.4906
UoGTrExpE1	0.2716	0.5073	0.4890	0.4643	0.5094	0.4641	0.5027	0.4778	0.4657
UDInfo_F_bm25	0.2708	0.5282	0.5606	0.5320	0.5289	0.5335	0.4810	0.5026	0.4983
ans_bm25	0.2706	0.5275	0.5601	0.5315	0.5282	0.5331	0.4801	0.5021	0.4978
UDInfo_F_mlp2	0.2703	0.5268	0.5597	0.5311	0.5269	0.5326	0.4803	0.5015	0.4976
FRT_attention	0.2702	0.5278	0.5841	0.5100	0.5281	0.5162	0.5213	0.5160	0.4881
UDInfo_F_lgbm2	0.2698	0.5261	0.5587	0.5304	0.5272	0.5320	0.4795	0.5006	0.4972
UDInfo_F_mlp4	0.2695	0.5251	0.5581	0.5294	0.5250	0.5309	0.4789	0.4999	0.4965
UDInfo_F_lgbm4	0.2693	0.5249	0.5574	0.5290	0.5263	0.5307	0.4791	0.4991	0.4962
FRT_diversity	0.2641	0.5195	0.5270	0.5020	0.5169	0.4998	0.4835	0.4861	0.4828
rmit_cidda_ir_5	0.2485	0.4788	0.5366	0.5021	0.4932	0.5038	0.4405	0.4738	0.4617
rmit_cidda_ir_1	0.2433	0.4383	0.5377	0.5064	0.4850	0.5101	0.4223	0.4774	0.4698
rmit_cidda_ir_4	0.2431	0.4444	0.5319	0.5008	0.4780	0.5052	0.4278	0.4700	0.4570
rmit_cidda_ir_7	0.2426	0.4321	0.5317	0.5012	0.4858	0.5052	0.4232	0.4718	0.4638
rmit_cidda_ir_3	0.2420	0.4302	0.5297	0.4994	0.4846	0.5037	0.4208	0.4707	0.4636
rmit_cidda_ir_6	0.2418	0.4312	0.5278	0.4969	0.4845	0.5015	0.4218	0.4686	0.4613
rmit_cidda_ir_8	0.2415	0.4315	0.5259	0.4949	0.4844	0.4996	0.4219	0.4671	0.4600
rmit_cidda_ir_2	0.2345	0.4122	0.5134	0.4829	0.4805	0.4875	0.4169	0.4552	0.4473

Table 2: Task 1 M_1 on individual fairness dimensions.

	Overall	2021	Internal	Demographic
tmt5	**0.3626**	**0.6034**	**0.5699**	**0.5389**
UoGRelvOnlyT1	0.3254	0.4757	0.5154	0.4338
UoGTrT1ColPRF	0.3254	0.4757	0.5154	0.4338
UoGTrExpE2	0.3230	0.4874	0.5160	0.4395
0mt5	0.2990	0.5143	0.4787	0.4563
0mt5_p	0.2949	0.5019	0.4702	0.4450
tmt5_p	0.2946	0.4987	0.4722	0.4428
FRT_constraint	0.2782	0.4691	0.4704	0.4152
bm25_p	0.2773	0.4716	0.4408	0.4181
UoGTrQE	0.2734	0.4411	0.4564	0.3866
UoGTrExpE1	0.2716	0.4208	0.4277	0.3828
UDInfo_F_bm25	0.2708	0.4666	0.4294	0.4145
ans_bm25	0.2706	0.4660	0.4285	0.4141
UDInfo_F_mlp2	0.2703	0.4660	0.4284	0.4138
FRT_attention	0.2702	0.4362	0.4420	0.3877
UDInfo_F_lgbm2	0.2698	0.4656	0.4276	0.4133
UDInfo_F_mlp4	0.2695	0.4649	0.4273	0.4127
UDInfo_F_lgbm4	0.2693	0.4645	0.4270	0.4123
FRT_diversity	0.2641	0.4515	0.4382	0.4012
rmit_cidda_ir_5	0.2485	0.4300	0.3834	0.3818
rmit_cidda_ir_1	0.2433	0.4375	0.3568	0.3899
rmit_cidda_ir_4	0.2431	0.4270	0.3603	0.3824
rmit_cidda_ir_7	0.2426	0.4327	0.3547	0.3868
rmit_cidda_ir_3	0.2420	0.4320	0.3527	0.3866
rmit_cidda_ir_6	0.2418	0.4297	0.3531	0.3851
rmit_cidda_ir_8	0.2415	0.4284	0.3533	0.3840
rmit_cidda_ir_2	0.2345	0.4160	0.3394	0.3735

Table 3: Task 1 M_1 on subsets of the fairness dimensions.

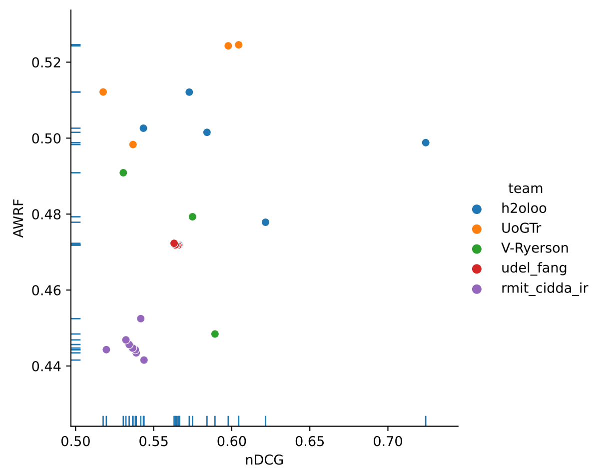

Figure 1: Task 1 submissions by individual component metrics (NDCG and AWRF). Higher values are better for both metrics.

- BM25 ranking from pyserini and pre-traned BERT for semantic score and re-ranked to fit target distribution at each ranking position by using greedy diversification, providing higher attention to protected group, and ensuring fairness in each position.
- Relevance ranking using BM25 from pyserini and LambdaMART lerning-to-rank model and a multi-layer-perception to re-rank.
- BM25 ranking from pyserini and weighted Reciprocal Ranking Fusion to diversify the ranking.
- Relevance-only approaches.

Table 1 shows the submitted systems ranked by the official Task 1 metric M_1 and its component parts nDCG and AWRF. Figure 1 plots the runs with the component metrics on the x and y axes. Unlike last year, we see less clustering of approaches from individual teams: two teams approaches had similar performance, while others are more scattered throughout the space.

We also computed fairness on individual dimension (Table 2 and Figure 2), and on the three subsets identified in Section 3.4 (Table 3 and Figure 3). For Task 1 the best-performing system overall also performed best on each individual category and subset; however, ordering of other systems changed between subsets or categories.

5.2 Task 2: Wikipedia Editors (Multiple Rankings)

Approaches for Task 2 included:

- Multi armed bandit strategies to select rankings from a pool of rankings considering each fairness category and observing the exposure and fairness-relevance relationship.
- Epsilon-greedy with weighted ranking, epsilon-decay strategy, and randomisation are used in ranking selection process.
- Relevance-only ranking.

Table 4 shows the submitted systems ranked by the official Task 2 metric EE-L and its component parts EE-D and EE-R. Figure 4 plots the runs with the component metrics on the x and y axes. Overall, the submitted systems generally performed better for one of the component metrics than they did for the other.

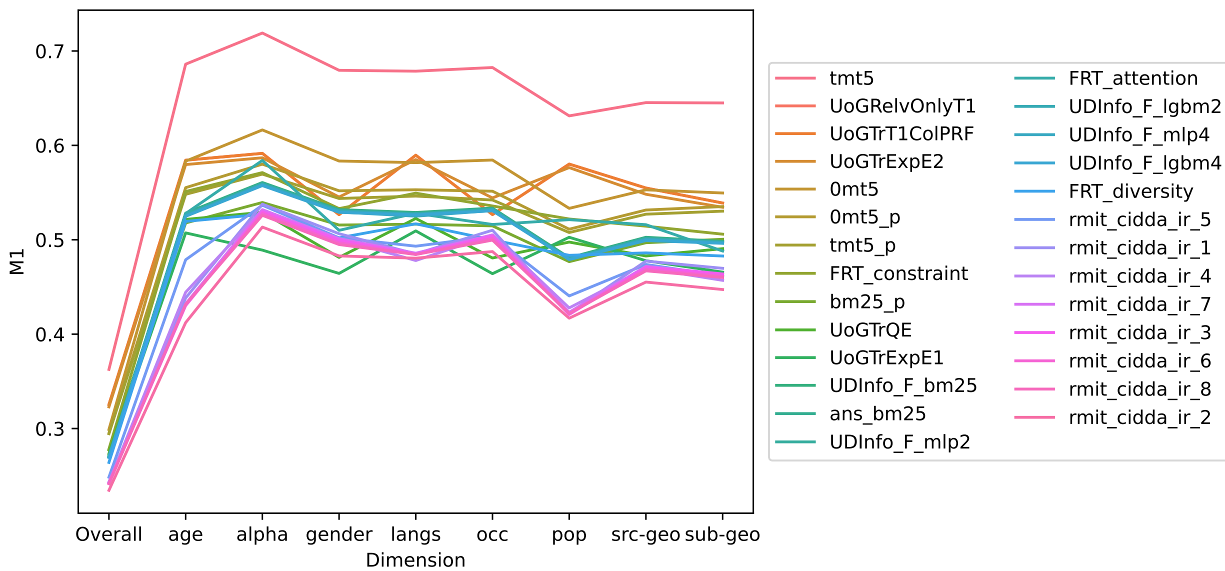

Figure 2: Task 1 M_1 on individual fairness dimensions.

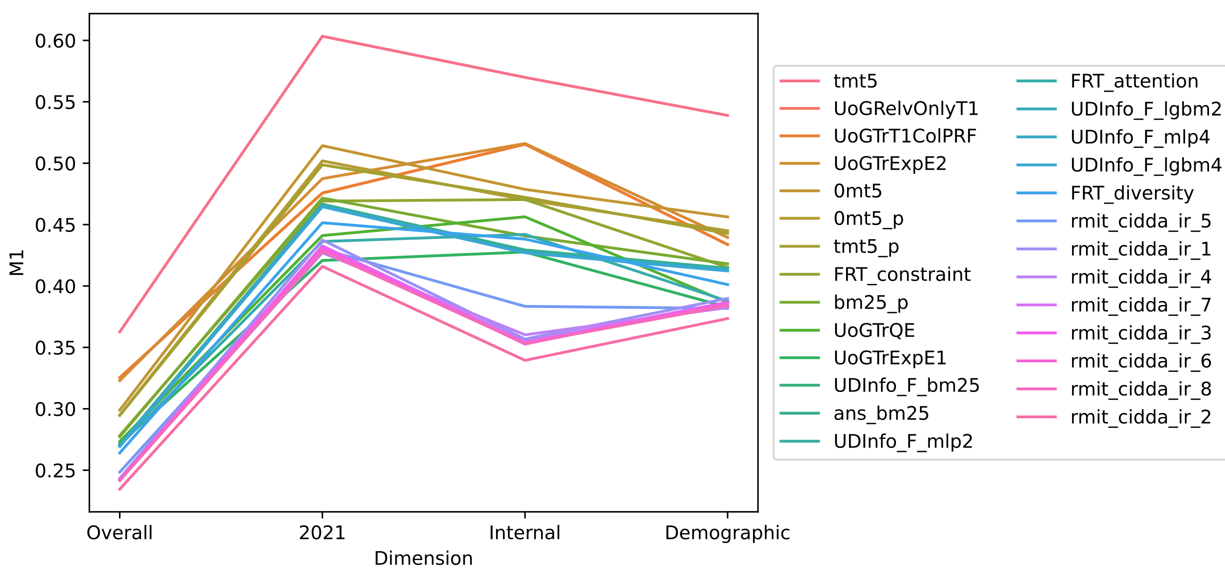

Figure 3: Task 1 M_1 on subsets of the fairness dimensions.

	EE-R	EE-D	EE-L	EE-L 95% CI
UoGTrMabWeSA	0.0485	1.0791	1.1231	(0.9790, 1.3096)
UoGTrMabSaWR	0.0512	1.1462	1.1847	(1.0258, 1.3739)
UoGTrMabSAED	0.0558	1.1935	1.2228	(1.0507, 1.4185)
tmt5_p_e	0.0934	1.3803	1.3345	(1.1409, 1.5793)
0mt5_p_e	0.1002	1.8563	1.7968	(1.5600, 2.0779)
bm25_p_e	0.0914	1.9077	1.8659	(1.5343, 2.2897)
UoGTrMabSaNR	0.0249	2.0486	2.1397	(1.8286, 2.6452)
UogTRelvOnlyT2	0.0788	2.2398	2.2231	(1.8645, 2.7786)
0mt5_e	0.0518	2.9483	2.9856	(2.5022, 3.7271)
tmt5_e	0.1116	3.4819	3.3997	(3.0171, 3.8588)
ans_bm25_e	0.0685	4.2286	4.2324	(2.7795, 8.3580)

Table 4: Task 2 runs. Lower EE-L is better. Confidence intervals are bootstrapped.

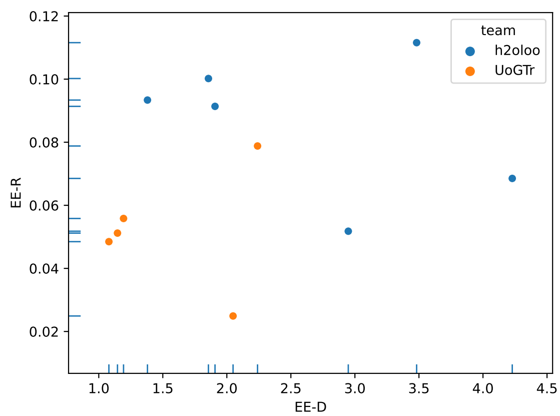

Figure 4: Task 2 submissions by expected exposure subcomponents. Lower EE-D is better; higher EE-R is better.

We also computed fairness on individual dimension (Table 5 and Figure 5), and on the three subsets identified in Section 3.4 (Table 6 and Figure 6). Unlike Task 1, we see more difference in fairness between different single attributes and subsets.

6 Limitations

The data and metrics in this task address a few specific types of unfairness, and do so partially. This is fundamentally true of any fairness intervention, and does not in any way diminish the value of the effort — it is impossible for any data set, task definition, or metric to fully capture fairness in a universal way, and all data and analyses have limitations.

Some of the limitations of the data and task include:

- **Fairness criteria**
  - **Gender**: For each Wikipedia article, we ascertain whether it is a biography, and, if so, which gender identity can be associated with the person it is about.[6] This data is directly determined

[6] Code: `https://github.com/geohci/miscellaneous-wikimedia/blob/master/wikidata-properties-spark/wikidata_gender_information.ipynb`

	Overall	age	alpha	gender	langs	occ	pop	src-geo	sub-geo
UoGTrMabWeSA	**1.123**	7.215	6.798	**15.163**	8.756	**15.127**	22.197	**4.798**	**5.925**
UoGTrMabSaWR	1.185	7.194	7.259	15.291	8.978	15.226	22.674	4.979	6.303
UoGTrMabSAED	1.223	**7.136**	7.111	15.636	9.127	15.551	**21.423**	4.884	6.314
tmt5_p_e	1.334	21.024	6.554	18.282	23.936	16.420	31.123	11.142	14.070
0mt5_p_e	1.797	23.243	**6.266**	18.579	24.876	16.516	37.273	11.962	16.320
bm25_p_e	1.866	25.273	7.607	20.687	26.782	18.361	39.964	13.278	16.995
UoGTrMabSaNR	2.140	8.637	9.343	15.981	**8.640**	16.078	42.181	5.894	7.790
UogTRelvOnlyT2	2.223	11.152	11.037	19.964	11.686	19.738	22.193	6.136	9.654
0mt5_e	2.986	27.765	11.149	22.881	27.967	20.988	43.410	14.288	19.946
tmt5_e	3.400	28.021	10.255	35.293	35.676	32.521	38.150	19.642	23.982
ans_bm25_e	4.232	29.440	14.830	27.922	32.793	25.886	50.670	16.335	25.838

Table 5: Task 2 EE-L on individual fairness dimensions.

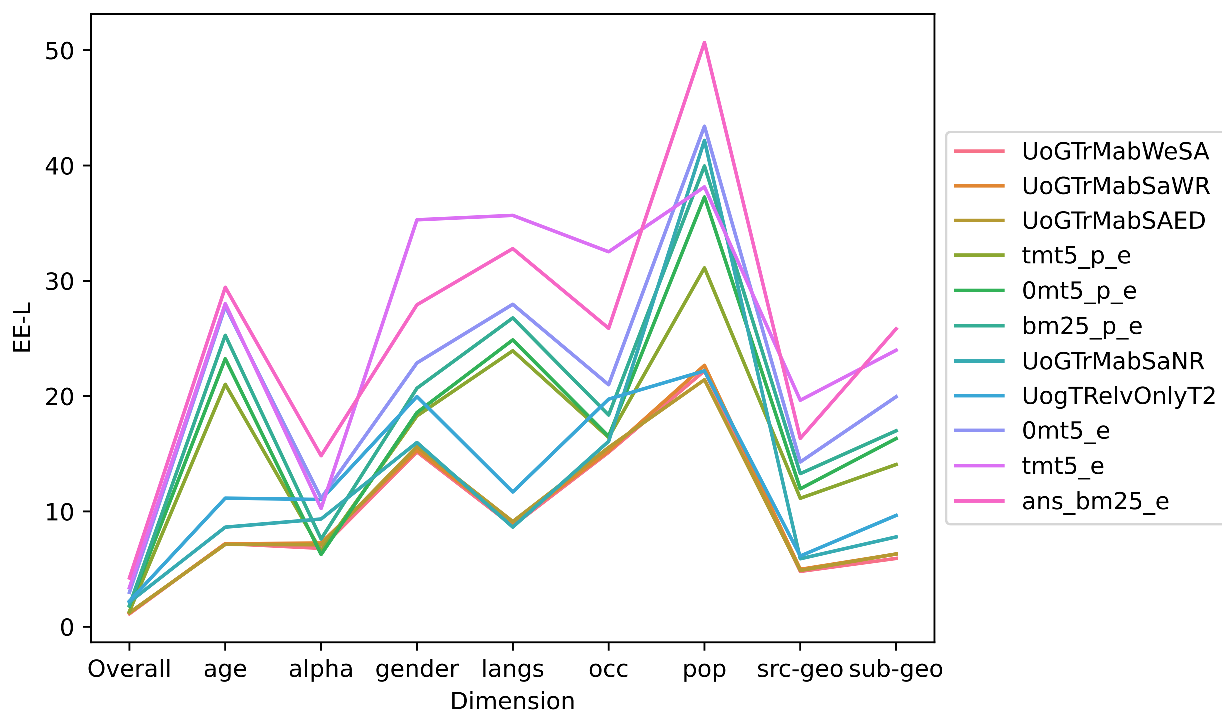

Figure 5: Task 2 EE-L on subsets of the fairness dimensions.

	Overall	2021	Internal	Demographic
UoGTrMabWeSA	**1.123**	**7.732**	**2.719**	**3.230**
UoGTrMabSaWR	1.185	8.276	2.936	3.373
UoGTrMabSAED	1.223	8.156	2.904	3.458
tmt5_p_e	1.334	14.856	5.185	7.021
0mt5_p_e	1.797	17.995	6.190	8.120
bm25_p_e	1.866	18.532	6.580	8.829
UoGTrMabSaNR	2.140	9.783	4.842	4.885
UogTRelvOnlyT2	2.223	12.157	4.667	5.989
0mt5_e	2.986	21.622	9.334	9.679
tmt5_e	3.400	28.262	8.391	13.300
ans_bm25_e	4.232	27.213	12.777	12.242

Table 6: Task 2 EE-L on subsets of the fairness dimensions.

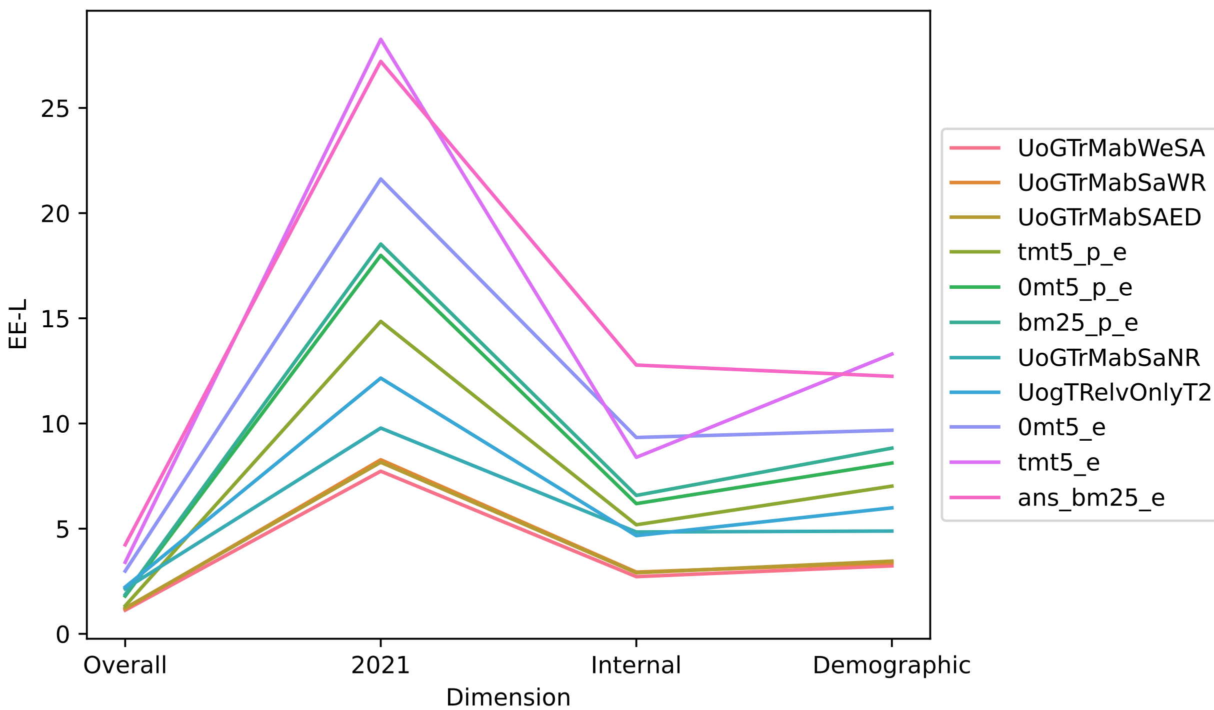

Figure 6: Task 2 EE-L on subsets of the fairness dimensions.

via Wikidata based on the instance-of property indicating the article is about a human (P31:Q5 in Wikidata terms) and then collecting the value associated with the sex-or-gender property (P21). Coverage here is quite high at 99.98% of biographies on Wikipedia having associated gender data on Wikidata.

Assigning gender identities to people is not a process without errors, biases, and ethical concerns. Applying the taxonomy developed by Pinney et al. [5] to this work yields the following summary: the primary referent of the gender data is the subject; we do use a gender variable and it's binary+other; gender determination is done via annotators (see details below); the gender data is used to measure bias and the goal is to audit system behavior. The process for assigning gender (annotation) is subject to some community-defined technical limitations[7] and the Wikidata policy on living people[8]. While a separate project, English Wikipedia's policies on gender identity[9] likely inform how many editors handle gender; in particular, this policy explicitly favors the most recent reliably-sourced *self-identification* for gender, so misgendering a biography subject is a violation of Wikipedia policy; there may be erroneous data, but such data seems to be a violation of policy instead of a policy decision. Wikidata:WikiProject LGBT has documented some clear limitations of gender data on Wikidata and a list of further discussions and considerations.[10] Since we are using gender data to calculate aggregate statistics, we judged these limitations to be less problematic than it would be if we were making decisions about individuals.

In our analysis (see Appendix A), we handle nonbinary gender identities by using four gender categories: unknown, male, female, and third.

We advise great care when working with the gender data, particularly outside the immediate context of the TREC task (either its original instance or using the data to evaluate comparable systems).

– **Geography**: For each Wikipedia article, we also ascertained which, if any, countries and continents are relevant to the content.[11] This was determined by directly looking up several community-maintained Wikidata structured data statements about the article. These properties were checked for the presence of countries, which were then mapped to continents via the United Nation's geoscheme.[12] While this data must meet Wikidata's verifiability guidelines,[13] it does suffer from varying levels of incompleteness. For example, only 74% of people on Wikidata have a country of citizenship property.[14] Furthermore, structured data is itself limited—e.g., country of citizenship does not appropriately capture people who are considered stateless though these people may have many strong ties to a country. It is not easy to evaluate whether this data is missing at random or biased against certain regions of the world. Care should be taken when interpreting the absence of associated continents in the data. Further details can be found in the code repository.[15]

  We also identify the associated countries and continents with the sources in the article. Each source is mapped to a country based on the URL or publisher associated with it. These mappings are built via a mixture of Wikidata, country extraction from whois records, and heuristics related to the top-level domain of the URL.[16] This is the only inferred attribute used that is not maintained by Wikimedians and thus is much more likely to contain errors. Because this data was also incomplete, we had the assessors annotate an additional 15,000 items to help add to the data and better understand its quality. The feedback from assessors was that it was a difficult task—i.e. inferring country information for a generic website or publisher is often not easy and can be quite

[7] https://www.wikidata.org/wiki/Property_talk:P21#Documentation
[8] https://www.wikidata.org/wiki/Wikidata:Living_people
[9] https://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style/Gender_identity
[10] https://www.wikidata.org/wiki/Wikidata:WikiProject_LGBT/gender
[11] Code: https://github.com/geohci/wiki-region-groundtruth/blob/main/wiki-region-data.ipynb
[12] https://en.wikipedia.org/wiki/United_Nations_geoscheme
[13] https://www.wikidata.org/wiki/Wikidata:Verifiability
[14] https://humaniki.wmcloud.org/gender-by-country
[15] https://github.com/geohci/wiki-region-groundtruth
[16] See code for more details: https://github.com/geohci/geo-provenance

ambiguous at times. While most items were only assessed once, 101 publishers received multiple assessments with 85 (84%) of these in agreement and 32 URLs received multiple assessments with 26 (81%) of these in agreement. We also had a few items for which we had already inferred regions that we had the assessors check: only 6 publishers were checked but 5 (83%) were in agreement and 90 URLs were checked with 82 (91%) in agreement. While this leaves uncertainty about the publisher data, it does suggest that the URL data is reasonable quality because a few of those in disagreement appear to be assessor errors.

  - **Age**: We calculate the associated age of the subject of the article in a similar manner to article topic geography (extracting dates from several pre-determined properties on Wikidata).[17] While geography is a multi-label feature, age is mapped to a single value via the median of the associated years and that median value is bucketed as pre-1900s, 20th century, or 21st century (and beyond). Like geography, it is not clear how many articles should have associated categories—e.g., many articles like those for plant species, do not clearly map to any specific time period—but it is safe to assume all biographies should have associated year data and we see 91.5% coverage suggesting relatively complete data.
  - **Occupation**: For each Wikipedia biography, we also ascertained which occupations could be associated with the person it is about. This data is directly determined via Wikidata by collecting the values associated with the occupation property (P106). For each occupation value, we then mapped it to one of 32 higher-order occupations based on the occupation ontology (using P279, the sub-class of, property for each occupation)[18]. The 32 higher-order occupations were hand-selected to give sufficient detail while remaining a manageable number of categories. On English Wikipedia, 92.1% of biographies have at least one associated occupation value that could be mapped to the 32 higher-order occupations.
  - **Popularity**: For each article, we calculated how many pageviews it received in February 2022. These pageview counts are based on webrequest logs[19] and filter out views from user-agents that explicitly identify themselves as spiders[20] and actors (shared user-agent and IP address) that seem to be automated in that they view more than 800 pages per hour[21]. These heuristics are not perfect however and traffic can be easily miscategorized if, for example, automated requests come from many different IPs or devices or actual users share a proxy that gives them the same IP and user-agent. The raw counts of pageviews were then converted into relative values between 0 and 1 by square-root transforming the value and normalizing to the 99th percentile of pageviews. Finally, these values were bucketed as [0 - 0.125), [0.125 - 0.250), [0.25 - 0.5), [0.5 - 1].
  - **Sitelinks**: The Wikimedia editor community maintains article sitelinks, or interlanguage links—i.e. explicit connection of articles about the same subject across language editions—via Wikidata. Almost all Wikipedia articles (99.92% for English)[22] have a corresponding Wikidata item and editors work to merge Wikidata items that are about the same subject so the sitelinks are aligned. Though there is no empirical data, it is generally accepted that most articles are appropriately linked to their corresponding other-language equivalents, especially in languages with shared scripts where simple approaches such as searching for an article title is often sufficient to identify matches.
  - **Other**: several fairness criteria are relatively straightforward and thus do not have many attached limitations. Specifically, the first letter of the article title (Alphabetical) and age of the article.

- **Relevance Criteria**

[17] See the code for more details: `https://gitlab.wikimedia.org/isaacj/miscellaneous-wikimedia/-/blob/master/wikidata-properties-spark/article-age.ipynb`

[18] For more details, see the code: `https://gitlab.wikimedia.org/isaacj/miscellaneous-wikimedia/-/blob/master/wikidata-properties-spark/wikidata_occupation_taxonomy.ipynb`

[19] `https://wikitech.wikimedia.org/wiki/Analytics/Data_Lake/Traffic/Webrequest`

[20] See: `https://meta.wikimedia.org/wiki/Research:Page_view`

[21] `https://wikitech.wikimedia.org/wiki/Analytics/Data_Lake/Traffic/BotDetection`

[22] `https://wikidata-analytics.wmcloud.org/app/WD_percentUsageDashboard`

- **WikiProject Relevance**: For the training queries, relevance was obtained from page lists for existing WikiProjects. While WikiProjects have broad coverage of English Wikipedia and we selected for WikiProjects that had tagged new articles in the recent months in the training data as a proxy for activity, it is certain that almost all WikiProjects are incomplete in tagging relevant content (itself a strong motivation for this task). While it is not easy to measure just how incomplete they are, it should not be assumed that content that has not been tagged as relevant to a WikiProject in the training data is indeed irrelevant.[23]
- **Work-needed**: Our proxy for work-needed is a coarse proxy. It is based on just a few simple features (page length, sections, images, categories, links, and references) and does not reflect the nuances of the work needed to craft a top-quality Wikipedia article.[24] A fully-fledged system for supporting Wikiprojects would also include a more nuanced approach to understanding the work needed for each article and how to appropriately allocate this work.

- **Task Definition**
  - **Existing Article Bias**: The task is limited to topics for which English Wikipedia already has articles. These tasks are not able to counteract biases in the processes by which articles come to exist (or are deleted [9])—recommending articles that should exist but don't is an interesting area for future study.
  - **Fairness constructs**: we focus on several fairness constructs in this challenge as metrics for which there is high data coverage and a clear mechanism for which "unfair" coverage might arise. That does not mean these are the most important constructs, but others—e.g., religion, sexuality, culture, race—generally are either more challenging to model or map to fairness goals [7].

References

[1] F. Diaz, B. Mitra, M. D. Ekstrand, A. J. Biega, and B. Carterette. Evaluating stochastic rankings with expected exposure. In *Proc. CIKM '20*, 2020. URL `https://arxiv.org/abs/2004.13157`.

[2] M. Grootendorst. Keybert: Minimal keyword extraction with bert., 2020. URL `https://doi.org/10.5281/zenodo.4461265`.

[3] S. Mitchell, E. Potash, S. Barocas, A. D'Amour, and K. Lum. Algorithmic fairness: Choices, assumptions, and definitions. *Annual Review of Statistics and Its Application*, 8, Nov. 2020. doi: 10.1146/annurev-statistics-042720-125902. URL `https://www.annualreviews.org/doi/abs/10.1146/annurev-statistics-042720-125902`.

[4] D. Pedreshi, S. Ruggieri, and F. Turini. Discrimination-aware data mining. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 560–568, 2008.

[5] C. Pinney, A. Raj, A. Hanna, and M. D. Ekstrand. Much ado about gender: Current practices and future recommendations for appropriate gender-aware information access. *arXiv preprint arXiv:2301.04780*, 2023.

[6] A. Raj, C. Wood, A. Montoly, and M. D. Ekstrand. Comparing fair ranking metrics. Sept. 2020. URL `http://arxiv.org/abs/2009.01311`.

[7] M. Redi, M. Gerlach, I. Johnson, J. Morgan, and L. Zia. A taxonomy of knowledge gaps for wikimedia projects (second draft). *arXiv preprint arXiv:2008.12314*, 2020.

[23] Current Wikiproject tags were extracted from the database tables maintained by the PageAssessments extension: `https://www.mediawiki.org/wiki/Extension:PageAssessments`

[24] For further details, see: `https://meta.wikimedia.org/wiki/Research:Prioritization_of_Wikipedia_Articles/Language-Agnostic_Quality#V2`

[8] P. Sapiezynski, W. Zeng, R. E Robertson, A. Mislove, and C. Wilson. Quantifying the impact of user attentionon fair group representation in ranked lists. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 553–562, 2019.

[9] F. Tripodi. Ms. categorized: Gender, notability, and inequality on wikipedia. *New Media & Society*, page 14614448211023772, 2021.

A Page Alignments

This notebook computes the *page alignments* from the Wikipedia metadata. These are then used by the task-specific alignment notebooks to compute target distributions and page alignment subsets for retrieved pages.

Warning: this notebook takes quite a bit of memory to run.

A.1 Setup

We begin by loading necessary libraries:

```
import sys
from pathlib import Path
import pandas as pd
import xarray as xr
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import gzip
import json
from natural.size import binarysize
```

Set up progress bar and logging support:

```
from tqdm.auto import tqdm
tqdm.pandas(leave=False)
```

```
import sys, logging
logging.basicConfig(level=logging.INFO, stream=sys.stderr)
log = logging.getLogger('PageAlignments')
```

And set up an output directory:

```
from wptrec.save import OutRepo
output = OutRepo('data/metric-tables')
```

A.2 Loading Data

Now we need to load the data.

A.2.1 Static Data

We need a set of subregions that are folded into Oceania:

```
oc_regions = [
    'Australia and New Zealand',
    'Melanesia',
    'Micronesia',
    'Polynesia',
]
```

And finally a name for unknown:

```
UNKNOWN = '@UNKNOWN'
```

Now all our background data is set up.

A.2.2 Page Data

Finally, we load the page metadata. This is a little manual to manage memory usage. Two memory usage tricks:

- Only import the things we need
- Use `sys.intern` for strings representing categoricals to decrease memory use

Bonus is that, through careful logic, we get a progress bar.

```
# META_FILE_TAG = 'discrete'
META_FILE_TAG = 'discrete_assessed'

page_path = Path(f'data/trec_2022_articles_{META_FILE_TAG}.json.gz')
page_file_size = page_path.stat().st_size
binarysize(page_file_size)
```

```
'238.76 MiB'
```

Definitions Let's define the different attributes we need to extract:

```
SUB_GEO_ATTR = 'page_subcont_regions'
SRC_GEO_ATTR = 'source_subcont_regions'
GENDER_ATTR = 'gender'
OCC_ATTR = 'occupations'
BASIC_ATTRS = [
    'page_id',
    'first_letter_category',
    'creation_date_category',
    'relative_pageviews_category',
    'num_sitelinks_category',
]
```

Read Data Now, we're going to process by creating lists we can reassemble with `pd.DataFrame.from_records`. We'll fill these with tuples and dictionaries as appropriate.

```
qual_recs = []
sub_geo_recs = []
src_geo_recs = []
gender_recs = []
occ_recs = []
att_recs = []
seen_pages = set()
```

And we're off.

```python
with tqdm(total=page_file_size, desc='compressed input', unit='B', unit_scale=True) as fpb:
    with open(page_path, 'rb') as gzf, gzip.GzipFile(fileobj=gzf, mode='r') as decoded:
        for line in decoded:
            line = json.loads(line)
            page = line['page_id']
            if page in seen_pages:
                continue
            else:
                seen_pages.add(page)

            # page quality
            qual_recs.append((page, line['qual_cat']))

            # page geography
            for geo in line[SUB_GEO_ATTR]:
                sub_geo_recs.append((page, sys.intern(geo)))

            # src geography
            psg = {'page_id': page}
            for g, v in line[SRC_GEO_ATTR].items():
                if g == 'UNK':
                    g = UNKNOWN
                psg[sys.intern(g)] = v
            src_geo_recs.append(psg)

            # genders
            for g in line[GENDER_ATTR]:
                gender_recs.append((page, sys.intern(g)))

            # occupations
            for occ in line[OCC_ATTR]:
                occ_recs.append((page, sys.intern(occ)))

            # other attributes
            att_recs.append(tuple((sys.intern(line[a]) if isinstance(line[a], str) else line[a])
                                  for a in BASIC_ATTRS))

            fpb.update(gzf.tell() - fpb.n)  # update the progress bar
```

{"model_id":"7a8ed81f35ca4fa0b50c58c43638f3e0","version_major":2,"version_minor":0}

Reassemble DFs Now we will assemble these records into data frames.

```python
quality = pd.DataFrame.from_records(qual_recs, columns=['page_id', 'quality'])
```

```python
sub_geo = pd.DataFrame.from_records(sub_geo_recs, columns=['page_id', 'sub_geo'])
sub_geo.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3773443 entries, 0 to 3773442
Data columns (total 2 columns):
```

```
 #   Column   Dtype
---  ------   -----
 0   page_id  int64
 1   sub_geo  object
dtypes: int64(1), object(1)
memory usage: 57.6+ MB
```

```
src_geo = pd.DataFrame.from_records(src_geo_recs)
src_geo.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6460210 entries, 0 to 6460209
Data columns (total 25 columns):
 #   Column                     Dtype
---  ------                     -----
 0   page_id                    int64
 1   Northern America           float64
 2   @UNKNOWN                   float64
 3   Northern Europe            float64
 4   Western Asia               float64
 5   Western Europe             float64
 6   Western Africa             float64
 7   Southern Europe            float64
 8   Australia and New Zealand  float64
 9   Central America            float64
 10  Eastern Asia               float64
 11  South America              float64
 12  Eastern Europe             float64
 13  Northern Africa            float64
 14  Eastern Africa             float64
 15  Southern Asia              float64
 16  Polynesia                  float64
 17  South-eastern Asia         float64
 18  Central Asia               float64
 19  Caribbean                  float64
 20  Southern Africa            float64
 21  Middle Africa              float64
 22  Antarctica                 float64
 23  Melanesia                  float64
 24  Micronesia                 float64
dtypes: float64(24), int64(1)
memory usage: 1.2 GB
```

```
gender = pd.DataFrame.from_records(gender_recs, columns=['page_id', 'gender'])
gender.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1850219 entries, 0 to 1850218
Data columns (total 2 columns):
 #   Column   Dtype
---  ------   -----
 0   page_id  int64
 1   gender   object
```

```
dtypes: int64(1), object(1)
memory usage: 28.2+ MB
```

```
occupations = pd.DataFrame.from_records(occ_recs, columns=['page_id', 'occ'])
occupations.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2445899 entries, 0 to 2445898
Data columns (total 2 columns):
 #   Column   Dtype
---  ------   -----
 0   page_id  int64
 1   occ      object
dtypes: int64(1), object(1)
memory usage: 37.3+ MB
```

```
cat_attrs = pd.DataFrame.from_records(att_recs, columns=BASIC_ATTRS)
cat_attrs.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6460210 entries, 0 to 6460209
Data columns (total 5 columns):
 #   Column                        Dtype
---  ------                        -----
 0   page_id                       int64
 1   first_letter_category         object
 2   creation_date_category        object
 3   relative_pageviews_category   object
 4   num_sitelinks_category        object
dtypes: int64(1), object(4)
memory usage: 246.4+ MB
```

```
all_pages = np.array(list(seen_pages))
all_pages = np.sort(all_pages)
all_pages = pd.Series(all_pages)
```

```
del src_geo_recs, sub_geo_recs
del gender_recs, occ_recs
del seen_pages
```

```
%reset -f out
```

```
Flushing output cache (1 entries)
```

```
import gc
gc.collect()
```

```
0
```

A.3 Helper Functions

These functions will help with further computations.

A.3.1 Normalize Distribution

We are going to compute a number of data frames that are alignment vectors, such that each row is to be a multinomial distribution. This function normalizes such a frame.

```
def norm_align_matrix(df):
    df = df.fillna(0)
    sums = df.sum(axis='columns')
    return df.div(sums, axis='rows')
```

A.4 Page Alignments

All of our metrics require page "alignments": the protected-group membership of each page.

A.4.1 Quality

Quality isn't an alignment, but we're going to save it here:

```
output.save_table(quality, 'page-quality', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\page-quality.csv.gz
INFO:wptrec.save:data\metric-tables\page-quality.csv.gz: 35.62 MiB
INFO:wptrec.save:saving Parquet to data\metric-tables\page-quality.parquet
INFO:wptrec.save:data\metric-tables\page-quality.parquet: 9.14 MiB
```

A.4.2 Page Geography

Let's start with the straight page geography alignment for the public evaluation of the training queries. We've already loaded it above.

We need to do a little cleanup on this data:

- Align pages with no known geography with '@UNKNOWN' (to sort before known categories)
- Replace Oceania subregions with Oceania

```
sub_geo.head()
```

```
   page_id          sub_geo
0      303  Northern America
1      307  Northern America
2      316  Northern America
3      324  Northern America
4      330   Southern Europe
```

Let's start by turning this into a wide frame:

```
sub_geo_align = sub_geo.assign(x=1).pivot(index='page_id', columns='sub_geo', values='x')
sub_geo_align.fillna(0, inplace=True)
sub_geo_align.head()
```

```
sub_geo  Antarctica  Australia and New Zealand  Caribbean  Central America  \
page_id
303             0.0                        0.0        0.0              0.0
307             0.0                        0.0        0.0              0.0
316             0.0                        0.0        0.0              0.0
324             0.0                        0.0        0.0              0.0
```

```
330             0.0                         0.0        0.0               0.0

sub_geo  Central Asia  Eastern Africa  Eastern Asia  Eastern Europe  \
page_id
303               0.0             0.0           0.0             0.0
307               0.0             0.0           0.0             0.0
316               0.0             0.0           0.0             0.0
324               0.0             0.0           0.0             0.0
330               0.0             0.0           0.0             0.0

sub_geo  Melanesia  Micronesia  ...  Northern Europe  Polynesia  \
page_id                         ...
303            0.0         0.0  ...              0.0        0.0
307            0.0         0.0  ...              0.0        0.0
316            0.0         0.0  ...              0.0        0.0
324            0.0         0.0  ...              0.0        0.0
330            0.0         0.0  ...              0.0        0.0

sub_geo  South America  South-eastern Asia  Southern Africa  Southern Asia  \
page_id
303                0.0                 0.0              0.0            0.0
307                0.0                 0.0              0.0            0.0
316                0.0                 0.0              0.0            0.0
324                0.0                 0.0              0.0            0.0
330                0.0                 0.0              0.0            0.0

sub_geo  Southern Europe  Western Africa  Western Asia  Western Europe
page_id
303                  0.0             0.0           0.0             0.0
307                  0.0             0.0           0.0             0.0
316                  0.0             0.0           0.0             0.0
324                  0.0             0.0           0.0             0.0
330                  1.0             0.0           0.0             0.0

[5 rows x 23 columns]
```

Now we need to collapse Oceania into one column.

```
ocean = sub_geo_align.loc[:, oc_regions].sum(axis='columns')
sub_geo_align = sub_geo_align.drop(columns=oc_regions)
sub_geo_align['Oceania'] = ocean
```

Next we need to add the Unknown column and expand this.
Sum the items to find total amounts, and then create a series for unknown:

```
sub_geo_sums = sub_geo_align.sum(axis='columns')
sub_geo_unknown = ~(sub_geo_sums > 0)
sub_geo_unknown = sub_geo_unknown.astype('f8')
sub_geo_unknown = sub_geo_unknown.reindex(all_pages, fill_value=1)
```

Now let's join this with the original frame:

```
sub_geo_align = sub_geo_unknown.to_frame(UNKNOWN).join(sub_geo_align, how='left')
sub_geo_align = norm_align_matrix(sub_geo_align)
sub_geo_align.head()
```

```
     @UNKNOWN  Antarctica  Caribbean  Central America  Central Asia  \
12        1.0         0.0        0.0              0.0           0.0
25        1.0         0.0        0.0              0.0           0.0
39        1.0         0.0        0.0              0.0           0.0
290       1.0         0.0        0.0              0.0           0.0
303       0.0         0.0        0.0              0.0           0.0

     Eastern Africa  Eastern Asia  Eastern Europe  Middle Africa  \
12              0.0           0.0             0.0            0.0
25              0.0           0.0             0.0            0.0
39              0.0           0.0             0.0            0.0
290             0.0           0.0             0.0            0.0
303             0.0           0.0             0.0            0.0

     Northern Africa  ...  Northern Europe  South America  South-eastern Asia  \
12               0.0  ...              0.0            0.0                 0.0
25               0.0  ...              0.0            0.0                 0.0
39               0.0  ...              0.0            0.0                 0.0
290              0.0  ...              0.0            0.0                 0.0
303              0.0  ...              0.0            0.0                 0.0

     Southern Africa  Southern Asia  Southern Europe  Western Africa  \
12               0.0            0.0              0.0             0.0
25               0.0            0.0              0.0             0.0
39               0.0            0.0              0.0             0.0
290              0.0            0.0              0.0             0.0
303              0.0            0.0              0.0             0.0

     Western Asia  Western Europe  Oceania
12            0.0             0.0      0.0
25            0.0             0.0      0.0
39            0.0             0.0      0.0
290           0.0             0.0      0.0
303           0.0             0.0      0.0

[5 rows x 21 columns]
```

```python
sub_geo_align.sort_index(axis='columns', inplace=True)
sub_geo_align.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6460210 entries, 12 to 70194530
Data columns (total 21 columns):
 #   Column              Dtype
---  ------              -----
 0   @UNKNOWN            float64
 1   Antarctica          float64
 2   Caribbean           float64
 3   Central America     float64
 4   Central Asia        float64
 5   Eastern Africa      float64
 6   Eastern Asia        float64
```

```
 7   Eastern Europe      float64
 8   Middle Africa       float64
 9   Northern Africa     float64
 10  Northern America    float64
 11  Northern Europe     float64
 12  Oceania             float64
 13  South America       float64
 14  South-eastern Asia  float64
 15  Southern Africa     float64
 16  Southern Asia       float64
 17  Southern Europe     float64
 18  Western Africa      float64
 19  Western Asia        float64
 20  Western Europe      float64
dtypes: float64(21)
memory usage: 1.3 GB
```

And convert this to an xarray for multidimensional usage:

```
sub_geo_xr = xr.DataArray(sub_geo_align, dims=['page', 'sub_geo'])
sub_geo_xr
```

```
<xarray.DataArray (page: 6460210, sub_geo: 21)>
array([[1., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       ...,
       [1., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.]])
Coordinates:
  * page     (page) int64 12 25 39 290 ... 70194480 70194481 70194489 70194530
  * sub_geo  (sub_geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
```

```
binarysize(sub_geo_xr.nbytes)
```

```
'1.90 GiB'
```

```
output.save_table(sub_geo_align, 'page-sub-geo-align', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\page-sub-geo-align.csv.gz
INFO:wptrec.save:data\metric-tables\page-sub-geo-align.csv.gz: 23.97 MiB
INFO:wptrec.save:saving Parquet to data\metric-tables\page-sub-geo-align.parquet
INFO:wptrec.save:data\metric-tables\page-sub-geo-align.parquet: 13.20 MiB
```

A.4.3 Page Source Geography

We now need to do a similar setup for page source geography, which comes to us as a multinomial distribution already.

```
src_geo.head()
```

```
   page_id  Northern America  @UNKNOWN  Northern Europe  Western Asia  \
0       12              50.0      42.0             40.0           2.0
```

```
1       25                42.0     152.0              16.0            NaN
2       39                24.0      25.0               6.0            NaN
3      290                15.0      13.0               3.0            NaN
4      303               202.0      23.0               9.0            NaN

   Western Europe  Western Africa  Southern Europe  Australia and New Zealand  \
0             NaN             NaN              NaN                        NaN
1             3.0             2.0              NaN                        NaN
2             5.0             NaN              NaN                        NaN
3             1.0             NaN              NaN                        NaN
4             4.0             NaN              NaN                        NaN

   Central America  ...  Southern Asia  Polynesia  South-eastern Asia  \
0              NaN  ...            NaN        NaN                 NaN
1              NaN  ...            NaN        NaN                 NaN
2              NaN  ...            NaN        NaN                 NaN
3              NaN  ...            NaN        NaN                 NaN
4              NaN  ...            NaN        NaN                 NaN

   Central Asia  Caribbean  Southern Africa  Middle Africa  Antarctica  \
0           NaN        NaN              NaN            NaN         NaN
1           NaN        NaN              NaN            NaN         NaN
2           NaN        NaN              NaN            NaN         NaN
3           NaN        NaN              NaN            NaN         NaN
4           NaN        NaN              NaN            NaN         NaN

   Melanesia  Micronesia
0        NaN         NaN
1        NaN         NaN
2        NaN         NaN
3        NaN         NaN
4        NaN         NaN

[5 rows x 25 columns]
```

Set up the index:

```
src_geo.set_index('page_id', inplace=True)
```

Expand, then put 1 in UNKNOWN for everything that's missing:

```
src_geo_align = src_geo.reindex(all_pages, fill_value=0)
src_geo_align.loc[src_geo_align.sum('columns') == 0, UNKNOWN] = 1
src_geo_align
```

```
          Northern America  @UNKNOWN  Northern Europe  Western Asia  \
12                    50.0      42.0             40.0           2.0
25                    42.0     152.0             16.0           NaN
39                    24.0      25.0              6.0           NaN
290                   15.0      13.0              3.0           NaN
303                  202.0      23.0              9.0           NaN
...                    ...       ...              ...           ...
70194419               NaN       1.0              NaN           NaN
```

```
70194480               NaN       1.0              NaN           NaN
70194481               7.0       1.0              NaN           NaN
70194489               NaN       2.0              NaN           NaN
70194530               8.0       NaN              NaN           NaN

          Western Europe  Western Africa  Southern Europe  \
12                   NaN             NaN              NaN
25                   3.0             2.0              NaN
39                   5.0             NaN              NaN
290                  1.0             NaN              NaN
303                  4.0             NaN              NaN
...                  ...             ...              ...
70194419             NaN             NaN              NaN
70194480             NaN             NaN              NaN
70194481             NaN             NaN              NaN
70194489             NaN             NaN              NaN
70194530             NaN             NaN              NaN

          Australia and New Zealand  Central America  Eastern Asia  ...  \
12                              NaN              NaN           NaN  ...
25                              NaN              NaN           NaN  ...
39                              NaN              NaN           NaN  ...
290                             NaN              NaN           NaN  ...
303                             NaN              NaN           NaN  ...
...                             ...              ...           ...  ...
70194419                        NaN              NaN           NaN  ...
70194480                        NaN              NaN           NaN  ...
70194481                        NaN              NaN           NaN  ...
70194489                        1.0              NaN           NaN  ...
70194530                        NaN              NaN           NaN  ...

          Southern Asia  Polynesia  South-eastern Asia  Central Asia  \
12                  NaN        NaN                 NaN           NaN
25                  NaN        NaN                 NaN           NaN
39                  NaN        NaN                 NaN           NaN
290                 NaN        NaN                 NaN           NaN
303                 NaN        NaN                 NaN           NaN
...                 ...        ...                 ...           ...
70194419            NaN        NaN                 NaN           NaN
70194480            NaN        NaN                 NaN           NaN
70194481            NaN        NaN                 NaN           NaN
70194489            NaN        NaN                 NaN           NaN
70194530            NaN        NaN                 NaN           NaN

          Caribbean  Southern Africa  Middle Africa  Antarctica  Melanesia  \
12              NaN              NaN            NaN         NaN        NaN
25              NaN              NaN            NaN         NaN        NaN
39              NaN              NaN            NaN         NaN        NaN
290             NaN              NaN            NaN         NaN        NaN
303             NaN              NaN            NaN         NaN        NaN
...             ...              ...            ...         ...        ...
```

```
70194419        NaN              NaN            NaN          NaN         NaN
70194480        NaN              NaN            NaN          NaN         NaN
70194481        NaN              NaN            NaN          NaN         NaN
70194489        NaN              NaN            NaN          NaN         NaN
70194530        NaN              NaN            NaN          NaN         NaN

          Micronesia
12               NaN
25               NaN
39               NaN
290              NaN
303              NaN
...              ...
70194419         NaN
70194480         NaN
70194481         NaN
70194489         NaN
70194530         NaN

[6460210 rows x 24 columns]
```

Collapse Oceania:

```
ocean = src_geo_align.loc[:, oc_regions].sum(axis='columns')
src_geo_align = src_geo_align.drop(columns=oc_regions)
src_geo_align['Oceania'] = ocean
```

And normalize.

```
src_geo_align = norm_align_matrix(src_geo_align)

src_geo_align.sort_index(axis='columns', inplace=True)
src_geo_align.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6460210 entries, 12 to 70194530
Data columns (total 21 columns):
 #   Column             Dtype
---  ------             -----
 0   @UNKNOWN           float64
 1   Antarctica         float64
 2   Caribbean          float64
 3   Central America    float64
 4   Central Asia       float64
 5   Eastern Africa     float64
 6   Eastern Asia       float64
 7   Eastern Europe     float64
 8   Middle Africa      float64
 9   Northern Africa    float64
 10  Northern America   float64
 11  Northern Europe    float64
 12  Oceania            float64
 13  South America      float64
```

```
 14  South-eastern Asia  float64
 15  Southern Africa     float64
 16  Southern Asia       float64
 17  Southern Europe     float64
 18  Western Africa      float64
 19  Western Asia        float64
 20  Western Europe      float64
dtypes: float64(21)
memory usage: 1.1 GB
```

Xarray:

```
src_geo_xr = xr.DataArray(src_geo_align, dims=['page', 'src_geo'])
src_geo_xr
```

```
<xarray.DataArray (page: 6460210, src_geo: 21)>
array([[0.31343284, 0.        , 0.        , ..., 0.        , 0.01492537,
        0.        ],
       [0.70697674, 0.        , 0.        , ..., 0.00930233, 0.        ,
        0.01395349],
       [0.41666667, 0.        , 0.        , ..., 0.        , 0.        ,
        0.08333333],
       ...,
       [0.125     , 0.        , 0.        , ..., 0.        , 0.        ,
        0.        ],
       [0.66666667, 0.        , 0.        , ..., 0.        , 0.        ,
        0.        ],
       [0.        , 0.        , 0.        , ..., 0.        , 0.        ,
        0.        ]])
Coordinates:
  * page     (page) int64 12 25 39 290 ... 70194480 70194481 70194489 70194530
  * src_geo  (src_geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
```

And save:

```
output.save_table(src_geo_align, 'page-src-geo-align', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\page-src-geo-align.csv.gz
INFO:wptrec.save:data\metric-tables\page-src-geo-align.csv.gz: 43.69 MiB
INFO:wptrec.save:saving Parquet to data\metric-tables\page-src-geo-align.parquet
INFO:wptrec.save:data\metric-tables\page-src-geo-align.parquet: 28.94 MiB
```

A.4.4 Gender

Now let's work on extracting gender - this is going work a lot like page geography.

```
gender.head()
```

```
   page_id  gender
0      307    male
1      308    male
2      339  female
3      340    male
4      344    male
```

And summarize:

```
gender['gender'].value_counts()
```

```
male                          1495445
female                         353301
transgender female                636
non-binary                        329
transgender male                  197
intersex                           94
eunuch                             70
genderfluid                        29
genderqueer                        27
cisgender female                   18
two-spiriit                        11
travesti                           10
transgender person                 10
cisgender male                      7
agender                             6
transmasculine                      6
neutral sex                         5
transfeminine                       4
bigender                            4
third gender                        2
demiboy                             2
fa'afafine                          2
neutrois                            1
assigned female at birth            1
māhū                                1
hijra                               1
Name: gender, dtype: int64
```

Now, we're going to do a little more work to reduce the dimensionality of the space. Points:

1. Trans men are men
2. Trans women are women
3. Cis/trans status is an adjective that can be dropped for the present purposes

The result is that we will collapse "transgender female" and "cisgender female" into "female".

The **downside** to this is that trans men are probabily significantly under-represented, but are now being collapsed into the dominant group.

```
pgcol = gender['gender']
pgcol = pgcol.str.replace(r'(?:tran|ci)sgender\s+((?:fe)?male)', r'\1', regex=True)
pgcol.value_counts()
```

```
male                          1495649
female                         353955
non-binary                        329
intersex                           94
eunuch                             70
genderfluid                        29
genderqueer                        27
```

```
two-spiriit                     11
transgender person              10
travesti                        10
agender                          6
transmasculine                   6
neutral sex                      5
transfeminine                    4
bigender                         4
third gender                     2
demiboy                          2
fa'afafine                       2
māhū                             1
hijra                            1
neutrois                         1
assigned female at birth         1
Name: gender, dtype: int64
```

Now, we're going to group the remaining gender identities together under the label 'NB'. As noted above, this is a debatable exercise that collapses a lot of identity.

```
gender_labels = [UNKNOWN, 'female', 'male', 'NB']
pgcol[~pgcol.isin(gender_labels)] = 'NB'
pgcol.value_counts()
```

```
male      1495649
female     353955
NB            615
Name: gender, dtype: int64
```

Now put this column back in the frame and deduplicate.

```
page_gender = gender.assign(gender=pgcol)
page_gender = page_gender.drop_duplicates()
```

```
del pgcol
```

Now we need to add unknown genders.

```
kg_mask = all_pages.isin(page_gender['page_id'])
unknown = all_pages[~kg_mask]
page_gender = pd.concat([
    page_gender,
    pd.DataFrame({'page_id': unknown, 'gender': UNKNOWN})
], ignore_index=True)
page_gender
```

```
          page_id    gender
0             307      male
1             308      male
2             339    female
3             340      male
4             344      male
...           ...       ...
6460607  70194419  @UNKNOWN
```

```
6460608  70194480  @UNKNOWN
6460609  70194481  @UNKNOWN
6460610  70194489  @UNKNOWN
6460611  70194530  @UNKNOWN

[6460612 rows x 2 columns]
```

And make an alignment matrix:

```
gender_align = page_gender.reset_index().assign(x=1).pivot(index='page_id', columns='gender', values='x')
gender_align.fillna(0, inplace=True)
gender_align = gender_align.reindex(columns=gender_labels)
gender_align.head()
```

```
gender   @UNKNOWN  female  male   NB
page_id
12            1.0     0.0   0.0  0.0
25            1.0     0.0   0.0  0.0
39            1.0     0.0   0.0  0.0
290           1.0     0.0   0.0  0.0
303           1.0     0.0   0.0  0.0
```

Let's see how frequent each of the genders is:

```
gender_align.sum(axis=0).sort_values(ascending=False)
```

```
gender
@UNKNOWN    4610461.0
male        1495647.0
female       353933.0
NB              571.0
dtype: float64
```

And convert to an xarray:

```
gender_xr = xr.DataArray(gender_align, dims=['page', 'gender'])
gender_xr
```

```
<xarray.DataArray (page: 6460210, gender: 4)>
array([[1., 0., 0., 0.],
       [1., 0., 0., 0.],
       [1., 0., 0., 0.],
       ...,
       [1., 0., 0., 0.],
       [1., 0., 0., 0.],
       [1., 0., 0., 0.]])
Coordinates:
  * page     (page) int64 12 25 39 290 ... 70194480 70194481 70194489 70194530
  * gender   (gender) object '@UNKNOWN' 'female' 'male' 'NB'
```

```
binarysize(gender_xr.nbytes)
```

```
'206.73 MiB'
```

```
output.save_table(gender_align, 'page-gender-align', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\page-gender-align.csv.gz
INFO:wptrec.save:data\metric-tables\page-gender-align.csv.gz: 18.80 MiB
INFO:wptrec.save:saving Parquet to data\metric-tables\page-gender-align.parquet
INFO:wptrec.save:data\metric-tables\page-gender-align.parquet: 9.33 MiB
```

A.4.5 Occupation

Occupation works like gender, but without the need for processing.

Convert to a matrix:

```
occ_align = occupations.assign(x=1).pivot(index='page_id', columns='occ', values='x')
occ_align.head()
```

```
occ      activist  agricultural worker  artist  athlete  biologist  \
page_id
307           NaN                  1.0     NaN      NaN        NaN
308           NaN                  NaN     NaN      NaN        1.0
339           NaN                  NaN     NaN      NaN        NaN
340           NaN                  NaN     NaN      NaN        NaN
344           NaN                  NaN     1.0      NaN        NaN

occ      businessperson  chemist  civil servant  clergyperson  \
page_id
307                 NaN      NaN            NaN           NaN
308                 NaN      NaN            NaN           NaN
339                 NaN      NaN            NaN           NaN
340                 1.0      NaN            NaN           NaN
344                 1.0      NaN            NaN           NaN

occ      computer scientist  ...  military personnel  musician  \
page_id                      ...
307                     NaN  ...                 1.0       NaN
308                     NaN  ...                 NaN       NaN
339                     NaN  ...                 NaN       NaN
340                     NaN  ...                 NaN       NaN
344                     NaN  ...                 NaN       NaN

occ      performing artist  physicist  politician  scientist  \
page_id
307                    NaN        NaN         1.0        NaN
308                    NaN        1.0         NaN        1.0
339                    NaN        NaN         NaN        NaN
340                    NaN        NaN         NaN        NaN
344                    NaN        NaN         NaN        NaN

occ      social scientist  sportsperson (non-athlete)  \
page_id
307                   NaN                         NaN
308                   NaN                         NaN
339                   NaN                         NaN
340                   NaN                         NaN
344                   NaN                         NaN
```

```
occ      transportation occupation  writer
page_id
307                            NaN     1.0
308                            NaN     1.0
339                            NaN     1.0
340                            NaN     NaN
344                            NaN     1.0

[5 rows x 32 columns]
```

Set up unknown and merge:

```
occ_unk = pd.Series(1.0, index=all_pages)
occ_unk.index.name = 'page_id'
occ_kmask = all_pages.isin(occ_align.index)
occ_kmask.index = all_pages
occ_unk[occ_kmask] = 0
occ_align = occ_unk.to_frame(UNKNOWN).join(occ_align, how='left')
occ_align = norm_align_matrix(occ_align)
occ_align.head()
```

```
         @UNKNOWN  activist  agricultural worker  artist  athlete  biologist  \
page_id
12            1.0       0.0                  0.0     0.0      0.0        0.0
25            1.0       0.0                  0.0     0.0      0.0        0.0
39            1.0       0.0                  0.0     0.0      0.0        0.0
290           1.0       0.0                  0.0     0.0      0.0        0.0
303           1.0       0.0                  0.0     0.0      0.0        0.0

         businessperson  chemist  civil servant  clergyperson  ...  \
page_id                                                        ...
12                  0.0      0.0            0.0           0.0  ...
25                  0.0      0.0            0.0           0.0  ...
39                  0.0      0.0            0.0           0.0  ...
290                 0.0      0.0            0.0           0.0  ...
303                 0.0      0.0            0.0           0.0  ...

         military personnel  musician  performing artist  physicist  \
page_id
12                      0.0       0.0                0.0        0.0
25                      0.0       0.0                0.0        0.0
39                      0.0       0.0                0.0        0.0
290                     0.0       0.0                0.0        0.0
303                     0.0       0.0                0.0        0.0

         politician  scientist  social scientist  sportsperson (non-athlete)  \
page_id
12              0.0        0.0               0.0                         0.0
25              0.0        0.0               0.0                         0.0
39              0.0        0.0               0.0                         0.0
290             0.0        0.0               0.0                         0.0
```

```
303              0.0        0.0                 0.0                           0.0

         transportation occupation  writer
page_id
12                              0.0     0.0
25                              0.0     0.0
39                              0.0     0.0
290                             0.0     0.0
303                             0.0     0.0

[5 rows x 33 columns]
```

```
occ_xr = xr.DataArray(occ_align, dims=['page', 'occ'])
occ_xr
```

```
<xarray.DataArray (page: 6460210, occ: 33)>
array([[1., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       ...,
       [1., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.],
       [1., 0., 0., ..., 0., 0., 0.]])
Coordinates:
  * page     (page) int64 12 25 39 290 ... 70194480 70194481 70194489 70194530
  * occ      (occ) object '@UNKNOWN' 'activist' ... 'writer'
```

And save:

```
output.save_table(occ_align, 'page-occ-align', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\page-occ-align.csv.gz
INFO:wptrec.save:data\metric-tables\page-occ-align.csv.gz: 26.18 MiB
INFO:wptrec.save:saving Parquet to data\metric-tables\page-occ-align.parquet
INFO:wptrec.save:data\metric-tables\page-occ-align.parquet: 12.67 MiB
```

A.4.6 Other Attributes

The other attributes don't require as much re-processing - they can be used as-is as categorical variables. Let's save!

```
pages = cat_attrs.set_index('page_id')
pages
```

```
         first_letter_category creation_date_category  \
page_id
12                         a-d              2001-2006
25                         a-d              2001-2006
39                         a-d              2001-2006
290                        a-d              2001-2006
303                        a-d              2001-2006
...                        ...                    ...
70194419                   l-r              2017-2022
```

```
70194480                  a-d             2017-2022
70194481                  a-d             2017-2022
70194489                  l-r             2017-2022
70194530                  a-d             2017-2022

         relative_pageviews_category num_sitelinks_category
page_id
12                              High           5+ languages
25                              High           5+ languages
39                              High           5+ languages
290                             High           5+ languages
303                             High           5+ languages
...                              ...                    ...
70194419                         Low          2-4 languages
70194480                         Low           English only
70194481                         Low           English only
70194489                         Low          2-4 languages
70194530                         Low           English only

[6460210 rows x 4 columns]
```

Now each of these needs to become another table. The `get_dummies` function is our friend.

```python
alpha_align = pd.get_dummies(pages['first_letter_category'])
```

```python
output.save_table(alpha_align, 'page-alpha-align', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\page-alpha-align.csv.gz
INFO:wptrec.save:data\metric-tables\page-alpha-align.csv.gz: 19.47 MiB
INFO:wptrec.save:saving Parquet to data\metric-tables\page-alpha-align.parquet
INFO:wptrec.save:data\metric-tables\page-alpha-align.parquet: 10.52 MiB
```

```python
alpha_xr = xr.DataArray(alpha_align, dims=['page', 'alpha'])
```

```python
age_align = pd.get_dummies(pages['creation_date_category'])
output.save_table(age_align, 'page-age-align', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\page-age-align.csv.gz
INFO:wptrec.save:data\metric-tables\page-age-align.csv.gz: 17.29 MiB
INFO:wptrec.save:saving Parquet to data\metric-tables\page-age-align.parquet
INFO:wptrec.save:data\metric-tables\page-age-align.parquet: 7.53 MiB
```

```python
age_xr = xr.DataArray(age_align, dims=['page', 'age'])
```

```python
pop_align = pd.get_dummies(pages['relative_pageviews_category'])
output.save_table(pop_align, 'page-pop-align', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\page-pop-align.csv.gz
INFO:wptrec.save:data\metric-tables\page-pop-align.csv.gz: 18.69 MiB
INFO:wptrec.save:saving Parquet to data\metric-tables\page-pop-align.parquet
INFO:wptrec.save:data\metric-tables\page-pop-align.parquet: 9.52 MiB
```

```python
pop_xr = xr.DataArray(pop_align, dims=['page', 'pop'])
```

```
langs_align = pd.get_dummies(pages['num_sitelinks_category'])
output.save_table(langs_align, 'page-langs-align', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\page-langs-align.csv.gz
INFO:wptrec.save:data\metric-tables\page-langs-align.csv.gz: 18.64 MiB
INFO:wptrec.save:saving Parquet to data\metric-tables\page-langs-align.parquet
INFO:wptrec.save:data\metric-tables\page-langs-align.parquet: 9.80 MiB
```

```
langs_xr = xr.DataArray(langs_align, dims=['page', 'langs'])
```

A.5 Working with Alignments

At this point, we have computed an alignment matrix for each of our attributes, and extracted the qrels.

We will use the data saved from this in separate notebooks to compute targets and alignments for tasks.

B Task 1 Alignment

This notebook computes the target distributions and retrieved page alignments for **Task 1**. It depends on the output of the PageAlignments notebook.

This notebook can be run in two modes: 'train', to process the training topics, and 'eval' for the eval topics.

```
DATA_MODE = 'eval'
```

B.1 Setup

We begin by loading necessary libraries:

```
import sys
import warnings
from collections import namedtuple
from functools import reduce
from itertools import product
import operator
from pathlib import Path
```

```
import pandas as pd
import xarray as xr
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import gzip
import json
from natural.size import binarysize
from natural.number import number
```

Set up progress bar and logging support:

```
from tqdm.auto import tqdm
tqdm.pandas(leave=False)
```

```
import sys, logging
logging.basicConfig(level=logging.INFO, stream=sys.stderr)
log = logging.getLogger('Task1Alignment')
```

And set up an output directory:

```
from wptrec.save import OutRepo
output = OutRepo('data/metric-tables')
```

B.2 Data and Helpers

Most data loading is outsourced to `MetricInputs`. First we save the data mode where metric inputs can find it:

```
import wptrec
wptrec.DATA_MODE = DATA_MODE
```

```
from MetricInputs import *
```

```
INFO:MetricInputs:reading data\metric-tables\page-sub-geo-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-src-geo-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-gender-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-occ-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-alpha-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-age-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-pop-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-langs-align.parquet
```

```
dimensions
```

```
[<dimension "sub-geo": 21 levels>,
 <dimension "src-geo": 21 levels>,
 <dimension "gender": 4 levels>,
 <dimension "occ": 33 levels>,
 <dimension "alpha": 4 levels>,
 <dimension "age": 4 levels>,
 <dimension "pop": 4 levels>,
 <dimension "langs": 3 levels>]
```

B.2.1 qrel join

We want a function to join alignments with qrels:

```
def qr_join(align):
    return qrels.join(align, on='page_id').set_index(['topic_id', 'page_id'])
```

B.2.2 norm_dist

And a function to normalize to a distribution:

```
def norm_dist_df(mat):
    sums = mat.sum('columns')
    return mat.divide(sums, 'rows')
```

B.3 Prep Overview

Now that we have our alignments and qrels, we are ready to prepare the Task 1 metrics.

We're first going to prepare the target distributions; then we will compute the alignments for the retrieved pages.

B.4 Subject Geography

Subject geography targets the average of the relevant set alignments and the world population.

```
qr_sub_geo_align = qr_join(sub_geo_align)
qr_sub_geo_align
```

```
                   @UNKNOWN  Antarctica  Caribbean  Central America  \
topic_id page_id
187      682            1.0         0.0        0.0              0.0
         954            0.0         0.0        0.0              0.0
         1170           1.0         0.0        0.0              0.0
         1315           1.0         0.0        0.0              0.0
         1322           0.0         0.0        0.0              0.0
...                     ...         ...        ...              ...
2872     69877511       1.0         0.0        0.0              0.0
         69878912       1.0         0.0        0.0              0.0
         69879322       1.0         0.0        0.0              0.0
         69881345       0.0         0.0        0.0              0.0
         69883661       1.0         0.0        0.0              0.0

                   Central Asia  Eastern Africa  Eastern Asia  Eastern Europe  \
topic_id page_id
187      682                0.0             0.0           0.0             0.0
         954                0.0             0.0           0.0             0.0
         1170               0.0             0.0           0.0             0.0
         1315               0.0             0.0           0.0             0.0
         1322               0.0             0.0           0.0             0.0
...                         ...             ...           ...             ...
2872     69877511           0.0             0.0           0.0             0.0
         69878912           0.0             0.0           0.0             0.0
         69879322           0.0             0.0           0.0             0.0
         69881345           0.0             0.0           0.0             0.0
         69883661           0.0             0.0           0.0             0.0

                   Middle Africa  Northern Africa  ...  Northern Europe  \
topic_id page_id                                   ...
187      682                 0.0              0.0  ...              0.0
         954                 0.0              0.0  ...              0.0
         1170                0.0              0.0  ...              0.0
         1315                0.0              0.0  ...              0.0
         1322                0.0              0.0  ...              0.0
...                          ...              ...  ...              ...
2872     69877511            0.0              0.0  ...              0.0
         69878912            0.0              0.0  ...              0.0
         69879322            0.0              0.0  ...              0.0
         69881345            0.0              0.0  ...              0.0
```

```
        69883661           0.0              0.0  ...             0.0

                 Oceania  South America  South-eastern Asia  \
topic_id page_id
187      682         0.0            0.0                 0.0
         954         0.0            0.0                 0.0
         1170        0.0            0.0                 0.0
         1315        0.0            0.0                 0.0
         1322        0.0            0.0                 0.0
...                  ...            ...                 ...
2872     69877511    0.0            0.0                 0.0
         69878912    0.0            0.0                 0.0
         69879322    0.0            0.0                 0.0
         69881345    0.0            0.0                 1.0
         69883661    0.0            0.0                 0.0

                 Southern Africa  Southern Asia  Southern Europe  \
topic_id page_id
187      682                 0.0            0.0              0.0
         954                 0.0            0.0              0.0
         1170                0.0            0.0              0.0
         1315                0.0            0.0              0.0
         1322                0.0            0.0              1.0
...                          ...            ...              ...
2872     69877511            0.0            0.0              0.0
         69878912            0.0            0.0              0.0
         69879322            0.0            0.0              0.0
         69881345            0.0            0.0              0.0
         69883661            0.0            0.0              0.0

                 Western Africa  Western Asia  Western Europe
topic_id page_id
187      682                0.0           0.0             0.0
         954                0.0           0.0             1.0
         1170               0.0           0.0             0.0
         1315               0.0           0.0             0.0
         1322               0.0           0.0             0.0
...                         ...           ...             ...
2872     69877511           0.0           0.0             0.0
         69878912           0.0           0.0             0.0
         69879322           0.0           0.0             0.0
         69881345           0.0           0.0             0.0
         69883661           0.0           0.0             0.0

[2737612 rows x 21 columns]
```

For purely geographic fairness, we just need to average the unknowns with the world pop:

```
qr_sub_geo_tgt = qr_sub_geo_align.groupby('topic_id').mean()
qr_sub_geo_fk = qr_sub_geo_tgt.iloc[:, 1:].sum('columns')
qr_sub_geo_tgt.iloc[:, 1:] *= 0.5
qr_sub_geo_tgt.iloc[:, 1:] += qr_sub_geo_fk.apply(lambda k: world_pop * k * 0.5)
qr_sub_geo_tgt.head()
```

```
          @UNKNOWN    Antarctica  Caribbean  Central America  Central Asia  \
topic_id
187       0.161757  6.472220e-08   0.004007         0.012384      0.004401
270       0.242805  5.846440e-08   0.017378         0.014851      0.005852
359       0.183666  6.303060e-08   0.017007         0.014391      0.003689
365       0.201370  6.166361e-08   0.007572         0.012774      0.004079
400       0.258172  5.727783e-08   0.004827         0.013104      0.003552

          Eastern Africa  Eastern Asia  Eastern Europe  Middle Africa  \
topic_id
187             0.022830      0.112412        0.033440       0.008264
270             0.037144      0.106411        0.053948       0.009914
359             0.021289      0.118833        0.017016       0.007747
365             0.022296      0.104172        0.035950       0.011613
400             0.020758      0.101462        0.027533       0.007496

          Northern Africa  ...  Northern Europe   Oceania  South America  \
topic_id                   ...
187              0.014711  ...         0.133172  0.020594       0.030093
270              0.017165  ...         0.058914  0.020977       0.038029
359              0.011968  ...         0.006663  0.005588       0.029521
365              0.015012  ...         0.029218  0.016421       0.036189
400              0.012440  ...         0.076621  0.023341       0.030668

          South-eastern Asia  Southern Africa  Southern Asia  Southern Europe  \
topic_id
187                 0.043274         0.004694       0.116350         0.059294
270                 0.038750         0.008852       0.101007         0.044103
359                 0.035681         0.003675       0.099904         0.010362
365                 0.053554         0.003956       0.100548         0.065794
400                 0.036634         0.005453       0.101073         0.027173

          Western Africa  Western Asia  Western Europe
topic_id
187             0.020306      0.023583        0.058312
270             0.026599      0.022927        0.055952
359             0.018935      0.014239        0.012154
365             0.024046      0.029213        0.031859
400             0.018965      0.018795        0.056502

[5 rows x 21 columns]
```

Make sure the rows are distributions:

```
qr_sub_geo_tgt.sum('columns').describe()
```

```
count    5.000000e+01
mean     1.000000e+00
std      1.409697e-16
min      1.000000e+00
25%      1.000000e+00
50%      1.000000e+00
```

```
75%      1.000000e+00
max      1.000000e+00
dtype: float64
```

Everything is 1, we're good to go!

```
output.save_table(qr_sub_geo_tgt, f'task1-{DATA_MODE}-sub-geo-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task1-eval-sub-geo-target.csv.gz
INFO:wptrec.save:data\metric-tables\task1-eval-sub-geo-target.csv.gz: 10.66 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task1-eval-sub-geo-target.parquet
INFO:wptrec.save:data\metric-tables\task1-eval-sub-geo-target.parquet: 25.97 KiB
```

B.5 Source Geography

Source geography works the same way.

```
qr_src_geo_align = qr_join(src_geo_align)
qr_src_geo_align
```

```
                   @UNKNOWN  Antarctica  Caribbean  Central America  \
topic_id page_id
187      682       0.400000         0.0        0.0              0.0
         954       0.257143         0.0        0.0              0.0
         1170      0.368421         0.0        0.0              0.0
         1315      0.375000         0.0        0.0              0.0
         1322      0.428571         0.0        0.0              0.0
...                     ...         ...        ...              ...
2872     69877511  1.000000         0.0        0.0              0.0
         69878912  0.366667         0.0        0.0              0.0
         69879322  0.200000         0.0        0.0              0.0
         69881345  0.500000         0.0        0.0              0.0
         69883661  0.000000         0.0        0.0              0.0

                   Central Asia  Eastern Africa  Eastern Asia  Eastern Europe  \
topic_id page_id
187      682                0.0             0.0           0.0             0.0
         954                0.0             0.0           0.0             0.0
         1170               0.0             0.0           0.0             0.0
         1315               0.0             0.0           0.0             0.0
         1322               0.0             0.0           0.0             0.0
...                         ...             ...           ...             ...
2872     69877511           0.0             0.0           0.0             0.0
         69878912           0.0             0.0           0.0             0.0
         69879322           0.0             0.0           0.0             0.0
         69881345           0.0             0.0           0.0             0.0
         69883661           0.0             0.0           0.0             0.0

                   Middle Africa  Northern Africa  ...  Northern Europe  \
topic_id page_id                                   ...
187      682                 0.0              0.0  ...         0.150000
         954                 0.0              0.0  ...         0.285714
         1170                0.0              0.0  ...         0.052632
```

```
         1315                0.0              0.0  ...        0.000000
         1322                0.0              0.0  ...        0.000000
...                          ...              ...  ...             ...
2872     69877511            0.0              0.0  ...        0.000000
         69878912            0.0              0.0  ...        0.000000
         69879322            0.0              0.0  ...        0.000000
         69881345            0.0              0.0  ...        0.000000
         69883661            0.0              0.0  ...        0.000000

                    Oceania  South America  South-eastern Asia  \
topic_id page_id
187      682       0.000000            0.0                 0.0
         954       0.000000            0.0                 0.0
         1170      0.052632            0.0                 0.0
         1315      0.000000            0.0                 0.0
         1322      0.000000            0.0                 0.0
...                     ...            ...                 ...
2872     69877511  0.000000            0.0                 0.0
         69878912  0.000000            0.0                 0.1
         69879322  0.000000            0.0                 0.0
         69881345  0.000000            0.0                 0.5
         69883661  0.000000            0.0                 0.0

                   Southern Africa  Southern Asia  Southern Europe  \
topic_id page_id
187      682                   0.0            0.0         0.000000
         954                   0.0            0.0         0.000000
         1170                  0.0            0.0         0.000000
         1315                  0.0            0.0         0.000000
         1322                  0.0            0.0         0.571429
...                            ...            ...              ...
2872     69877511              0.0            0.0         0.000000
         69878912              0.0            0.0         0.000000
         69879322              0.0            0.0         0.000000
         69881345              0.0            0.0         0.000000
         69883661              0.0            0.0         0.000000

                   Western Africa  Western Asia  Western Europe
topic_id page_id
187      682                  0.0         0.000        0.050000
         954                  0.0         0.000        0.171429
         1170                 0.0         0.000        0.000000
         1315                 0.0         0.125        0.000000
         1322                 0.0         0.000        0.000000
...                           ...           ...             ...
2872     69877511             0.0         0.000        0.000000
         69878912             0.0         0.000        0.000000
         69879322             0.0         0.600        0.000000
         69881345             0.0         0.000        0.000000
         69883661             0.0         0.000        0.000000
```

```
[2737612 rows x 21 columns]
```

And repeat:

```
qr_src_geo_tgt = qr_src_geo_align.groupby('topic_id').mean()
qr_src_geo_fk = qr_src_geo_tgt.iloc[:, 1:].sum('columns')
qr_src_geo_tgt.iloc[:, 1:] *= 0.5
qr_src_geo_tgt.iloc[:, 1:] += qr_src_geo_fk.apply(lambda k: world_pop * k * 0.5)
qr_src_geo_tgt.head()
```

```
          @UNKNOWN    Antarctica  Caribbean  Central America  Central Asia  \
topic_id
187       0.391787  4.696121e-08   0.002250         0.008070      0.002876
270       0.420047  4.477917e-08   0.003611         0.008171      0.002702
359       0.372489  4.845126e-08   0.003072         0.008260      0.002821
365       0.364985  4.903066e-08   0.010223         0.008492      0.002984
400       0.422769  2.798744e-07   0.002478         0.008311      0.002702

          Eastern Africa  Eastern Asia  Eastern Europe  Middle Africa  \
topic_id
187             0.016153      0.077938        0.019365       0.005790
270             0.015759      0.073673        0.019488       0.005524
359             0.016384      0.084042        0.013101       0.005947
365             0.017147      0.082518        0.018251       0.007674
400             0.015381      0.074893        0.018031       0.005497

          Northern Africa  ...  Northern Europe   Oceania  South America  \
topic_id                   ...
187              0.009195  ...         0.110871  0.011692       0.019483
270              0.008721  ...         0.044787  0.010542       0.020577
359              0.009209  ...         0.007908  0.003628       0.018333
365              0.009672  ...         0.021657  0.012542       0.020322
400              0.008827  ...         0.069702  0.019709       0.019562

          South-eastern Asia  Southern Africa  Southern Asia  Southern Europe  \
topic_id
187                 0.029280         0.003079       0.081422         0.019888
270                 0.026281         0.003505       0.073534         0.018938
359                 0.027301         0.002669       0.076759         0.007120
365                 0.038885         0.002730       0.078353         0.039960
400                 0.027291         0.003381       0.078346         0.015888

          Western Africa  Western Asia  Western Europe
topic_id
187             0.014278      0.013633        0.033541
270             0.013802      0.011568        0.061875
359             0.014524      0.010901        0.010185
365             0.015051      0.020196        0.029345
400             0.013821      0.012813        0.025499

[5 rows x 21 columns]
```

Make sure the rows are distributions:

```
qr_src_geo_tgt.sum('columns').describe()
```

```
count    5.000000e+01
mean     1.000000e+00
std      1.218255e-16
min      1.000000e+00
25%      1.000000e+00
50%      1.000000e+00
75%      1.000000e+00
max      1.000000e+00
dtype: float64
```

Everything is 1, we're good to go!

```
output.save_table(qr_src_geo_tgt, f'task1-{DATA_MODE}-src-geo-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task1-eval-src-geo-target.csv.gz
INFO:wptrec.save:data\metric-tables\task1-eval-src-geo-target.csv.gz: 10.64 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task1-eval-src-geo-target.parquet
INFO:wptrec.save:data\metric-tables\task1-eval-src-geo-target.parquet: 25.97 KiB
```

B.6 Gender

Now we're going to grab the gender alignments. Again, we ignore UNKNOWN.

```
qr_gender_align = qr_join(gender_align)
qr_gender_align.head()
```

```
                  @UNKNOWN  female  male   NB
topic_id page_id
187      682           1.0     0.0   0.0  0.0
         954           0.0     0.0   1.0  0.0
         1170          1.0     0.0   0.0  0.0
         1315          1.0     0.0   0.0  0.0
         1322          1.0     0.0   0.0  0.0
```

```
qr_gender_tgt = qr_gender_align.groupby('topic_id').mean()
qr_gender_fk = qr_gender_tgt.iloc[:, 1:].sum('columns')
qr_gender_tgt.iloc[:, 1:] *= 0.5
qr_gender_tgt.iloc[:, 1:] += qr_gender_fk.apply(lambda k: gender_tgt * k * 0.5)
qr_gender_tgt.head()
```

```
          @UNKNOWN    female      male        NB
topic_id
187       0.888195  0.033910  0.077336  0.000574
270       0.371833  0.257322  0.367774  0.003231
359       0.340156  0.170558  0.486007  0.003299
365       0.424643  0.183396  0.389116  0.002877
400       0.011697  0.408054  0.575302  0.005275
```

```
output.save_table(qr_gender_tgt, f'task1-{DATA_MODE}-gender-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task1-eval-gender-target.csv.gz
INFO:wptrec.save:data\metric-tables\task1-eval-gender-target.csv.gz: 2.22 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task1-eval-gender-target.parquet
INFO:wptrec.save:data\metric-tables\task1-eval-gender-target.parquet: 6.90 KiB
```

B.7 Remaining Attributes

The remaining attributes don't need any further processing, as they aren't averaged.

```
qr_occ_align = qr_join(occ_align)
qr_occ_tgt = qr_occ_align.groupby('topic_id').sum()
qr_occ_tgt = norm_dist_df(qr_occ_tgt)
qr_occ_tgt.head()
```

```
          @UNKNOWN  activist  agricultural worker    artist   athlete  \
topic_id
187       0.891108  0.000192             0.000049  0.005105  0.000383
270       0.379033  0.000143             0.000153  0.000569  0.597543
359       0.355009  0.000216             0.000048  0.000564  0.587417
365       0.427646  0.000081             0.000016  0.000186  0.499385
400       0.044346  0.004397             0.000387  0.316302  0.003669

          biologist  businessperson   chemist  civil servant  clergyperson  \
topic_id
187        0.000193        0.002763  0.000005       0.000194      0.000081
270        0.000145        0.001116  0.000123       0.000671      0.000110
359        0.000045        0.004931  0.000062       0.000336      0.000046
365        0.000023        0.001868  0.000047       0.000207      0.000094
400        0.001530        0.019926  0.000269       0.002284      0.001724

          ...  military personnel  musician  performing artist  physicist  \
topic_id  ...
187       ...            0.000335  0.000128           0.000110   0.000052
270       ...            0.000867  0.000404           0.001072   0.000024
359       ...            0.001501  0.000922           0.002827   0.000010
365       ...            0.000696  0.000274           0.001756   0.000000
400       ...            0.002074  0.010823           0.128105   0.000393

          politician  scientist  social scientist  sportsperson (non-athlete)  \
topic_id
187         0.001044   0.001168          0.000461                    0.000040
270         0.002388   0.000277          0.000275                    0.008550
359         0.001808   0.000037          0.000045                    0.031237
365         0.001031   0.000063          0.000070                    0.061864
400         0.007384   0.003000          0.003345                    0.001635

          transportation occupation    writer
topic_id
187                        0.000031  0.001421
270                        0.000281  0.000811
359                        0.000059  0.001414
365                        0.000094  0.000777
400                        0.000520  0.249432

[5 rows x 33 columns]
```

```
output.save_table(qr_occ_tgt, f'task1-{DATA_MODE}-occ-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task1-eval-occ-target.csv.gz
INFO:wptrec.save:data\metric-tables\task1-eval-occ-target.csv.gz: 14.99 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task1-eval-occ-target.parquet
INFO:wptrec.save:data\metric-tables\task1-eval-occ-target.parquet: 38.59 KiB
```

```
qr_age_align = qr_join(age_align)
qr_age_tgt = norm_dist_df(qr_age_align.groupby('topic_id').sum())
output.save_table(qr_age_tgt, f'task1-{DATA_MODE}-age-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task1-eval-age-target.csv.gz
INFO:wptrec.save:data\metric-tables\task1-eval-age-target.csv.gz: 2.13 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task1-eval-age-target.parquet
INFO:wptrec.save:data\metric-tables\task1-eval-age-target.parquet: 6.23 KiB
```

```
qr_alpha_align = qr_join(alpha_align)
qr_alpha_tgt = norm_dist_df(qr_alpha_align.groupby('topic_id').sum())
output.save_table(qr_alpha_tgt, f'task1-{DATA_MODE}-alpha-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task1-eval-alpha-target.csv.gz
INFO:wptrec.save:data\metric-tables\task1-eval-alpha-target.csv.gz: 2.11 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task1-eval-alpha-target.parquet
INFO:wptrec.save:data\metric-tables\task1-eval-alpha-target.parquet: 5.10 KiB
```

```
qr_langs_align = qr_join(langs_align)
qr_langs_tgt = norm_dist_df(qr_langs_align.groupby('topic_id').sum())
output.save_table(qr_langs_tgt, f'task1-{DATA_MODE}-langs-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task1-eval-langs-target.csv.gz
INFO:wptrec.save:data\metric-tables\task1-eval-langs-target.csv.gz: 1.67 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task1-eval-langs-target.parquet
INFO:wptrec.save:data\metric-tables\task1-eval-langs-target.parquet: 5.20 KiB
```

```
qr_pop_align = qr_join(pop_align)
qr_pop_tgt = norm_dist_df(qr_pop_align.groupby('topic_id').sum())
output.save_table(qr_pop_tgt, f'task1-{DATA_MODE}-pop-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task1-eval-pop-target.csv.gz
INFO:wptrec.save:data\metric-tables\task1-eval-pop-target.csv.gz: 2.17 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task1-eval-pop-target.parquet
INFO:wptrec.save:data\metric-tables\task1-eval-pop-target.parquet: 6.15 KiB
```

B.8 Multidimensional Alignment

Now, we need to set up the *multidimensional* alignment. The basic version is just to multiply the targets, but that doesn't include the target averaging we want to do for geographic and gender targets.

Doing that averaging further requires us to very carefully handle the unknown cases.

We are going to proceed in three steps:

1. Define the averaged dimensions (with their background targets) and the un-averaged dimensions
2. Demonstrate the logic by working through the alignment computations for a single topic
3. Apply step (2) to all topics

B.8.1 Dimension Definitions

Let's define background distributions for some of our dimensions:

```
dim_backgrounds = {
    'sub-geo': world_pop,
    'src-geo': world_pop,
    'gender': gender_tgt,
}
```

Now we'll make a list of dimensions to treat with averaging:

```
DR = namedtuple('DimRec', ['name', 'align', 'background'], defaults=[None])
avg_dims = [
    DR(d.name, d.page_align_xr, xr.DataArray(dim_backgrounds[d.name], dims=[d.name]))
    for d in dimensions
    if d.name in dim_backgrounds
]
[d.name for d in avg_dims]
```

```
['sub-geo', 'src-geo', 'gender']
```

And a list of dimensions to use as-is:

```
raw_dims = [
    DR(d.name, d.page_align_xr)
    for d in dimensions
    if d.name not in dim_backgrounds
]
[d.name for d in raw_dims]
```

```
['occ', 'alpha', 'age', 'pop', 'langs']
```

Now: these dimension are in the original order - `dimensions` has the averaged dimensions before the non-averaged ones. **This is critical for the rest of the code to work.**

B.8.2 Demo

To demonstrate how the logic works, let's first work it out in cells for one query (1).

What are its documents?

```
qno = qrels['topic_id'].iloc[0]
qdf = qrels[qrels['topic_id'] == qno]
qdf.name = qno
qdf
```

```
       topic_id   page_id
0           187       682
1           187       954
2           187      1170
3           187      1315
4           187      1322
...         ...       ...
68641       187  69882575
68642       187  69890514
```

```
68643       187  69891122
68644       187  69891390
68645       187  69892653

[68646 rows x 2 columns]
```

We can use these page IDs to get its alignments.

```python
q_pages = qdf['page_id'].values
```

Accumulating Initial Targets We're now going to grab the dimensions that have targets, and create a single xarray with all of them:

```python
q_xta = reduce(operator.mul, [d.align.loc[q_pages] for d in avg_dims])
q_xta
```

```
<xarray.DataArray (page: 68646, sub-geo: 21, src-geo: 21, gender: 4)>
array([[[[0.4       , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         ...,
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         [0.05      , 0.        , 0.        , 0.        ]],

        [[0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         ...,
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ]],

        [[0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         ...,
...
         ...,
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ]],

        [[0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         ...,
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ]],

        [[0.        , 0.        , 0.        , 0.        ],
```

```
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         ...,
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ],
         [0.        , 0.        , 0.        , 0.        ]]]])
Coordinates:
  * page     (page) int64 682 954 1170 1315 ... 69891122 69891390 69892653
  * sub-geo  (sub-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
  * src-geo  (src-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
  * gender   (gender) object '@UNKNOWN' 'female' 'male' 'NB'
```

We can similarly do this for the dimensions without targets:

```
q_raw_xta = reduce(operator.mul, [d.align.loc[q_pages] for d in raw_dims])
q_raw_xta
```

```
<xarray.DataArray (page: 68646, occ: 33, alpha: 4, age: 4, pop: 4, langs: 3)>
array([[[[[[0., 1., 0.],
           [0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.]],

          [[0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.]],

          [[0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.]],

          [[0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.]]],

...

         [[[0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.]],

          [[0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.]],

          [[0., 0., 0.],
           [0., 0., 0.],
```

```
           [0., 0., 0.],
           [0., 0., 0.]],

          [[0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.]]]]]]])
Coordinates:
  * page     (page) int64 682 954 1170 1315 ... 69891122 69891390 69892653
  * occ      (occ) object '@UNKNOWN' 'activist' ... 'writer'
  * alpha    (alpha) object 'a-d' 'e-k' 'l-r' 's-'
  * age      (age) object '2001-2006' '2007-2011' '2012-2016' '2017-2022'
  * pop      (pop) object 'High' 'Low' 'Medium-High' 'Medium-Low'
  * langs    (langs) object '2-4 languages' '5+ languages' 'English only'
```

Now, we need to combine this with the other matrix to produce a complete alignment matrix, which we then will collapse into a query target matrix. However, we don't have memory to do the whole thing at one go. Therefore, we will do it page by page.

The `mean_outer` function does this:

```
from wptrec.dimension import mean_outer

q_tam = mean_outer(q_xta, q_raw_xta)
q_tam
```

```
<xarray.DataArray (sub-geo: 21, src-geo: 21, gender: 4, occ: 33, alpha: 4,
                   age: 4, pop: 4, langs: 3)>
array([[[[[[[[3.90778732e-05, 9.13512756e-04, 0.00000000e+00],
             [1.07309246e-03, 1.09248385e-03, 8.45444299e-04],
             [2.37733808e-04, 9.16356065e-04, 5.09862192e-05],
             [3.91334003e-04, 3.40654865e-04, 1.97493559e-04]],

            [[8.32428068e-06, 4.32454542e-05, 2.61467791e-06],
             [4.90345161e-04, 3.17004794e-04, 7.68906390e-04],
             [7.80659667e-05, 1.96644900e-04, 1.78047115e-05],
             [2.89383842e-04, 2.64631783e-04, 3.21959246e-04]],

            [[2.03267319e-06, 7.67543443e-06, 0.00000000e+00],
             [3.34556557e-04, 1.53049653e-04, 5.17167629e-04],
             [4.49973617e-05, 2.80944473e-05, 2.21113706e-05],
             [8.65276618e-05, 7.67250106e-05, 1.43730437e-04]],

            [[2.70602185e-05, 4.39063929e-06, 7.07563858e-06],
             [2.58404302e-04, 9.37175266e-05, 7.87475979e-04],
             [9.01797073e-06, 1.61861013e-06, 4.82104549e-05],
             [1.18014878e-04, 2.01703724e-05, 9.15858957e-05]]],

...

           [[[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
             [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
             [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
             [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],
```

```
            [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
             [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
             [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
             [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

            [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
             [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
             [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
             [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

            [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
             [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
             [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
             [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]]]]]]]])
Coordinates:
  * sub-geo  (sub-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
  * src-geo  (src-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
  * gender   (gender) object '@UNKNOWN' 'female' 'male' 'NB'
  * occ      (occ) object '@UNKNOWN' 'activist' ... 'writer'
  * alpha    (alpha) object 'a-d' 'e-k' 'l-r' 's-'
  * age      (age) object '2001-2006' '2007-2011' '2012-2016' '2017-2022'
  * pop      (pop) object 'High' 'Low' 'Medium-High' 'Medium-Low'
  * langs    (langs) object '2-4 languages' '5+ languages' 'English only'
```

```
q_tam
```

```
<xarray.DataArray (sub-geo: 21, src-geo: 21, gender: 4, occ: 33, alpha: 4,
                   age: 4, pop: 4, langs: 3)>
array([[[[[[[[3.90778732e-05, 9.13512756e-04, 0.00000000e+00],
             [1.07309246e-03, 1.09248385e-03, 8.45444299e-04],
             [2.37733808e-04, 9.16356065e-04, 5.09862192e-05],
             [3.91334003e-04, 3.40654865e-04, 1.97493559e-04]],

            [[8.32428068e-06, 4.32454542e-05, 2.61467791e-06],
             [4.90345161e-04, 3.17004794e-04, 7.68906390e-04],
             [7.80659667e-05, 1.96644900e-04, 1.78047115e-05],
             [2.89383842e-04, 2.64631783e-04, 3.21959246e-04]],

            [[2.03267319e-06, 7.67543443e-06, 0.00000000e+00],
             [3.34556557e-04, 1.53049653e-04, 5.17167629e-04],
             [4.49973617e-05, 2.80944473e-05, 2.21113706e-05],
             [8.65276618e-05, 7.67250106e-05, 1.43730437e-04]],

            [[2.70602185e-05, 4.39063929e-06, 7.07563858e-06],
             [2.58404302e-04, 9.37175266e-05, 7.87475979e-04],
             [9.01797073e-06, 1.61861013e-06, 4.82104549e-05],
             [1.18014878e-04, 2.01703724e-05, 9.15858957e-05]]],

...

           [[[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
```

```
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

             [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

             [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

             [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]]]]]]]])
Coordinates:
  * sub-geo  (sub-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
  * src-geo  (src-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
  * gender   (gender) object '@UNKNOWN' 'female' 'male' 'NB'
  * occ      (occ) object '@UNKNOWN' 'activist' ... 'writer'
  * alpha    (alpha) object 'a-d' 'e-k' 'l-r' 's-'
  * age      (age) object '2001-2006' '2007-2011' '2012-2016' '2017-2022'
  * pop      (pop) object 'High' 'Low' 'Medium-High' 'Medium-Low'
  * langs    (langs) object '2-4 languages' '5+ languages' 'English only'
```

```
q_tam.sum()
```

```
<xarray.DataArray ()>
array(1.00001457)
```

In 2021, we ignored fully-unknown for Task 1. However, it isn't clear hot to properly do that with some attributes that are never fully unknown - they still need to be counted. Therefore, we consistently treat fully-unknown as a distinct category for both Task 1 and Task 2 metrics.

Data Subsetting Before we average, we need to be able to select data by its known/unknown status.

Let's start by making a list of cases - the known/unknown status of each dimension.

```
avg_cases = list(product(*[[True, False] for d in avg_dims]))
avg_cases
```

```
[(True, True, True),
 (True, True, False),
 (True, False, True),
 (True, False, False),
 (False, True, True),
 (False, True, False),
 (False, False, True),
 (False, False, False)]
```

The last entry is the all-unknown case - remove it:

```
avg_cases.pop()
avg_cases
```

```
[(True, True, True),
 (True, True, False),
 (True, False, True),
 (True, False, False),
 (False, True, True),
 (False, True, False),
 (False, False, True)]
```

We now want the ability to create an indexer to look up the subset of the alignment frame corresponding to a case. Let's write that function:

```
def case_selector(case):
    def mksel(known):
        if known:
            # select all but 1st column
            return slice(1, None, None)
        else:
            # select 1st column
            return 0

    return tuple(mksel(k) for k in case)
```

Let's test this function quick:

```
case_selector(avg_cases[0])
```

```
(slice(1, None, None), slice(1, None, None), slice(1, None, None))
```

```
case_selector(avg_cases[-1])
```

```
(0, 0, slice(1, None, None))
```

And make sure we can use it:

```
q_tam[case_selector(avg_cases[1])]
```

```
<xarray.DataArray (sub-geo: 20, src-geo: 20, occ: 33, alpha: 4, age: 4, pop: 4,
                   langs: 3)>
array([[[[[[[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

           [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

           [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
```

```
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

           [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]]],

...

          [[[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

           [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

           [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

           [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
            [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]]]]]]])
Coordinates:
  * sub-geo  (sub-geo) object 'Antarctica' 'Caribbean' ... 'Western Europe'
  * src-geo  (src-geo) object 'Antarctica' 'Caribbean' ... 'Western Europe'
    gender   <U8 '@UNKNOWN'
  * occ      (occ) object '@UNKNOWN' 'activist' ... 'writer'
  * alpha    (alpha) object 'a-d' 'e-k' 'l-r' 's-'
  * age      (age) object '2001-2006' '2007-2011' '2012-2016' '2017-2022'
  * pop      (pop) object 'High' 'Low' 'Medium-High' 'Medium-Low'
  * langs    (langs) object '2-4 languages' '5+ languages' 'English only'
```

Fantastic! Given a case (known and unknown statuses), we can select the subset of the target matrix with exactly those.

Averaging Ok, now we have to - very carefully - average with our target modifier. For each dimension that is not fully-unknown, we average with the intersectional target defined over the known dimensions.

At all times, we also need to respect the fraction of the total it represents.

We'll use the selection capabilities above to handle this.

First, let's make sure that our target matrix sums to 1 to start with:

```
q_tam.sum()
```

```
<xarray.DataArray ()>
array(1.00001457)
```

Fantastic. This means that if we sum up a subset of the data, it will give us the fraction of the distribution that has that combination of known/unknown status.

For each condition, we are going to proceed as follows:

1. Compute an appropriate intersectional background distribution (based on the dimensions that are "known")
2. Select the subset of the target matrix with this known status
3. Compute the sum of this subset
4. Re-normalize the subset to sum to 1
5. Compute a normalization table such that each coordinate in the distributions to correct sums to 1 (so multiplying this by the background distribution spreads the background across the other dimensions appropriately), and use this to spread the background distribution
6. Average with the spread background distribution
7. Re-normalize to preserve the original sum

Let's define the whole process as a function:

```
def avg_with_bg(tm, verbose=False):
    tm = tm.copy()

    tail_names = [d.name for d in raw_dims]

    # compute the tail mass for each coordinate (can be done once)
    tail_mass = tm.sum(tail_names)

    # now some things don't have any mass, but we still need to distribute background distributions.
    # solution: we impute the marginal tail distribution
    # first compute it
    tail_marg = tm.sum([d.name for d in avg_dims])
    # then impute that where we don't have mass
    tm_imputed = xr.where(tail_mass > 0, tm, tail_marg)
    # and re-compute the tail mass
    tail_mass = tm_imputed.sum(tail_names)
    # and finally we compute the rescaled matrix
    tail_scale = tm_imputed / tail_mass
    del tm_imputed

    for case in avg_cases:
        # for deugging: get names
        known_names = [d.name for (d, known) in zip(avg_dims, case) if known]
        if verbose:
            print('processing known:', known_names)

        # Step 1: background
        bg = reduce(operator.mul, [
            d.background
            for (d, known) in zip(avg_dims, case)
            if known
        ])
        if not np.allclose(bg.sum(), 1.0):
            warnings.warn('background distribution for {} sums to {}, expected 1'.format(known_names, bg
```

```python
        # Step 2: selector
        sel = case_selector(case)

        # Steps 3: sum in preparation for normalization
        c_sum = tm[sel].sum()

        # Step 5: spread the background
        bg_spread = bg * tail_scale[sel] * c_sum
        if not np.allclose(bg_spread.sum(), c_sum):
            warnings.warn('rescaled background sums to {}, expected c_sum'.format(bg_spread.values.sum()

        # Step 4 & 6: average with the background
        tm[sel] *= 0.5
        bg_spread *= 0.5
        tm[sel] += bg_spread

        if not np.allclose(tm[sel].sum(), c_sum):
            warnings.warn('target distribution for {} sums to {}, expected {}'.format(known_names, tm[se

    return tm
```

And apply it:

```python
q_target = avg_with_bg(q_tam, True)
q_target.sum()
```

```
processing known: ['sub-geo', 'src-geo', 'gender']
processing known: ['sub-geo', 'src-geo']
processing known: ['sub-geo', 'gender']
processing known: ['sub-geo']
processing known: ['src-geo', 'gender']
processing known: ['src-geo']
processing known: ['gender']
```

```
<xarray.DataArray ()>
array(1.00001457)
```

```python
q_target
```

```
<xarray.DataArray (sub-geo: 21, src-geo: 21, gender: 4, occ: 33, alpha: 4,
                   age: 4, pop: 4, langs: 3)>
array([[[[[[[[3.90778732e-05, 9.13512756e-04, 0.00000000e+00],
             [1.07309246e-03, 1.09248385e-03, 8.45444299e-04],
             [2.37733808e-04, 9.16356065e-04, 5.09862192e-05],
             [3.91334003e-04, 3.40654865e-04, 1.97493559e-04]],

            [[8.32428068e-06, 4.32454542e-05, 2.61467791e-06],
             [4.90345161e-04, 3.17004794e-04, 7.68906390e-04],
             [7.80659667e-05, 1.96644900e-04, 1.78047115e-05],
             [2.89383842e-04, 2.64631783e-04, 3.21959246e-04]],

            [[2.03267319e-06, 7.67543443e-06, 0.00000000e+00],
             [3.34556557e-04, 1.53049653e-04, 5.17167629e-04],
```

```
               [4.49973617e-05, 2.80944473e-05, 2.21113706e-05],
               [8.65276618e-05, 7.67250106e-05, 1.43730437e-04]],

              [[2.70602185e-05, 4.39063929e-06, 7.07563858e-06],
               [2.58404302e-04, 9.37175266e-05, 7.87475979e-04],
               [9.01797073e-06, 1.61861013e-06, 4.82104549e-05],
               [1.18014878e-04, 2.01703724e-05, 9.15858957e-05]]],

...

             [[[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
               [7.50104470e-13, 3.37547011e-12, 0.00000000e+00],
               [0.00000000e+00, 3.18794400e-12, 0.00000000e+00],
               [0.00000000e+00, 1.63415617e-12, 0.00000000e+00]],

              [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
               [7.76358126e-12, 2.32532386e-12, 2.25031341e-12],
               [0.00000000e+00, 4.50062682e-13, 0.00000000e+00],
               [7.50104470e-13, 0.00000000e+00, 0.00000000e+00]],

              [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
               [1.87526117e-12, 0.00000000e+00, 2.43783953e-12],
               [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
               [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

              [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
               [2.25031341e-12, 5.62578352e-13, 3.00041788e-12],
               [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
               [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]]]]]]]])
Coordinates:
  * sub-geo  (sub-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
  * src-geo  (src-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
  * gender   (gender) object '@UNKNOWN' 'female' 'male' 'NB'
  * occ      (occ) object '@UNKNOWN' 'activist' ... 'writer'
  * alpha    (alpha) object 'a-d' 'e-k' 'l-r' 's-'
  * age      (age) object '2001-2006' '2007-2011' '2012-2016' '2017-2022'
  * pop      (pop) object 'High' 'Low' 'Medium-High' 'Medium-Low'
  * langs    (langs) object '2-4 languages' '5+ languages' 'English only'
```

```
print(number(q_target.values.size), 'values taking', binarysize(q_target.nbytes))
```

```
11,176,704 values taking 89.41 MiB
```

Is it still a distribution?

```
q_target.sum()
```

```
<xarray.DataArray ()>
array(1.00001457)
```

We can unravel this value into a single-dimensional array representing the multidimensional target:

```
q_target.values.ravel()
```

```
array([3.90778732e-05, 9.13512756e-04, 0.00000000e+00, ...,
       0.00000000e+00, 0.00000000e+00, 0.00000000e+00])
```

Now we have all the pieces to compute this for each of our queries.

B.8.3 Implementing Function

To perform this combination for every query, we'll use a function that takes a data frame for a query's relevant docs and performs all of the above operations:

```python
def query_xalign(pages):
    # compute targets to average
    avg_pages = reduce(operator.mul, [d.align.loc[pages] for d in avg_dims])
    raw_pages = reduce(operator.mul, [d.align.loc[pages] for d in raw_dims])

    # convert to query distribution
    tgt = mean_outer(avg_pages, raw_pages)

    # average with background distributions
    tgt = avg_with_bg(tgt)

    # and return the result
    return tgt
```

Make sure it works:

```python
query_xalign(qdf.page_id.values)
```

```
<xarray.DataArray (sub-geo: 21, src-geo: 21, gender: 4, occ: 33, alpha: 4,
                   age: 4, pop: 4, langs: 3)>
array([[[[[[[[3.90778732e-05, 9.13512756e-04, 0.00000000e+00],
             [1.07309246e-03, 1.09248385e-03, 8.45444299e-04],
             [2.37733808e-04, 9.16356065e-04, 5.09862192e-05],
             [3.91334003e-04, 3.40654865e-04, 1.97493559e-04]],

            [[8.32428068e-06, 4.32454542e-05, 2.61467791e-06],
             [4.90345161e-04, 3.17004794e-04, 7.68906390e-04],
             [7.80659667e-05, 1.96644900e-04, 1.78047115e-05],
             [2.89383842e-04, 2.64631783e-04, 3.21959246e-04]],

            [[2.03267319e-06, 7.67543443e-06, 0.00000000e+00],
             [3.34556557e-04, 1.53049653e-04, 5.17167629e-04],
             [4.49973617e-05, 2.80944473e-05, 2.21113706e-05],
             [8.65276618e-05, 7.67250106e-05, 1.43730437e-04]],

            [[2.70602185e-05, 4.39063929e-06, 7.07563858e-06],
             [2.58404302e-04, 9.37175266e-05, 7.87475979e-04],
             [9.01797073e-06, 1.61861013e-06, 4.82104549e-05],
             [1.18014878e-04, 2.01703724e-05, 9.15858957e-05]]],

...

           [[[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
```

```
       [7.50104470e-13, 3.37547011e-12, 0.00000000e+00],
       [0.00000000e+00, 3.18794400e-12, 0.00000000e+00],
       [0.00000000e+00, 1.63415617e-12, 0.00000000e+00]],

      [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [7.76358126e-12, 2.32532386e-12, 2.25031341e-12],
       [0.00000000e+00, 4.50062682e-13, 0.00000000e+00],
       [7.50104470e-13, 0.00000000e+00, 0.00000000e+00]],

      [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [1.87526117e-12, 0.00000000e+00, 2.43783953e-12],
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

      [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [2.25031341e-12, 5.62578352e-13, 3.00041788e-12],
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]]]]]]]])
Coordinates:
  * sub-geo  (sub-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
  * src-geo  (src-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
  * gender   (gender) object '@UNKNOWN' 'female' 'male' 'NB'
  * occ      (occ) object '@UNKNOWN' 'activist' ... 'writer'
  * alpha    (alpha) object 'a-d' 'e-k' 'l-r' 's-'
  * age      (age) object '2001-2006' '2007-2011' '2012-2016' '2017-2022'
  * pop      (pop) object 'High' 'Low' 'Medium-High' 'Medium-Low'
  * langs    (langs) object '2-4 languages' '5+ languages' 'English only'
```

B.8.4 Computing Query Targets

Now with that function, we can compute the alignment vector for each query. Extract queries into a dictionary:

```python
queries = {
    t: df['page_id'].values
    for (t, df) in qrels.groupby('topic_id')
}
```

Make an index that we'll need later for setting up the XArray dimension:

```python
q_ids = pd.Index(queries.keys(), name='topic_id')
q_ids
```

```
Int64Index([ 187,  270,  359,  365,  400,  404,  480,  517,  568,  596,  715,
             807,  834,  881,  883,  949,  951,  955,  995, 1018, 1180, 1233,
            1328, 1406, 1417, 1448, 1449, 1479, 1499, 1548, 1558, 1647, 1685,
            1806, 1821, 1877, 1884, 1890, 2000, 2028, 2106, 2153, 2160, 2229,
            2244, 2448, 2483, 2758, 2867, 2872],
           dtype='int64', name='topic_id')
```

Now let's create targets for each of these:

```python
q_tgts = [query_xalign(queries[q]) for q in tqdm(q_ids)]
```

{"model_id":"d7cf659921754083b3c99d2a487c1f52","version_major":2,"version_minor":0}

Assemble a composite xarray:

```
q_tgts = xr.concat(q_tgts, q_ids)
q_tgts
```

```
<xarray.DataArray (topic_id: 50, sub-geo: 21, src-geo: 21, gender: 4, occ: 33,
                   alpha: 4, age: 4, pop: 4, langs: 3)>
array([[[[[[[[[3.90778732e-05, 9.13512756e-04, 0.00000000e+00],
              [1.07309246e-03, 1.09248385e-03, 8.45444299e-04],
              [2.37733808e-04, 9.16356065e-04, 5.09862192e-05],
              [3.91334003e-04, 3.40654865e-04, 1.97493559e-04]],

             [[8.32428068e-06, 4.32454542e-05, 2.61467791e-06],
              [4.90345161e-04, 3.17004794e-04, 7.68906390e-04],
              [7.80659667e-05, 1.96644900e-04, 1.78047115e-05],
              [2.89383842e-04, 2.64631783e-04, 3.21959246e-04]],

             [[2.03267319e-06, 7.67543443e-06, 0.00000000e+00],
              [3.34556557e-04, 1.53049653e-04, 5.17167629e-04],
              [4.49973617e-05, 2.80944473e-05, 2.21113706e-05],
              [8.65276618e-05, 7.67250106e-05, 1.43730437e-04]],

             [[2.70602185e-05, 4.39063929e-06, 7.07563858e-06],
              [2.58404302e-04, 9.37175266e-05, 7.87475979e-04],
              [9.01797073e-06, 1.61861013e-06, 4.82104549e-05],
              [1.18014878e-04, 2.01703724e-05, 9.15858957e-05]]],

...

            [[[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

             [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

             [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

             [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]]]]]]]]])
Coordinates:
  * sub-geo   (sub-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
```

```
  * src-geo   (src-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
  * gender    (gender) object '@UNKNOWN' 'female' 'male' 'NB'
  * occ       (occ) object '@UNKNOWN' 'activist' ... 'writer'
  * alpha     (alpha) object 'a-d' 'e-k' 'l-r' 's-'
  * age       (age) object '2001-2006' '2007-2011' '2012-2016' '2017-2022'
  * pop       (pop) object 'High' 'Low' 'Medium-High' 'Medium-Low'
  * langs     (langs) object '2-4 languages' '5+ languages' 'English only'
  * topic_id  (topic_id) int64 187 270 359 365 400 ... 2448 2483 2758 2867 2872
```

Save this to NetCDF (xarray's recommended format):

```
output.save_xarray(q_tgts, f'task1-{DATA_MODE}-int-targets')
```

```
INFO:wptrec.save:saving NetCDF to data\metric-tables\task1-eval-int-targets.nc
```

C Task 2 Alignment

This notebook computes the target distributions and retrieved page alignments for **Task 2**. It depends on the output of the PageAlignments notebook, as imported by MetricInputs.

This notebook can be run in two modes: 'train', to process the training topics, and 'eval' for the eval topics.

```
DATA_MODE = 'eval'
```

C.1 Setup

We begin by loading necessary libraries:

```
import sys
import operator
from functools import reduce
from itertools import product
from collections import namedtuple
from pathlib import Path
import pandas as pd
import xarray as xr
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import gzip
import json
from natural.size import binarysize
```

Set up progress bar and logging support:

```
from tqdm.auto import tqdm
tqdm.pandas(leave=False)
```

```
import sys, logging
logging.basicConfig(level=logging.INFO, stream=sys.stderr)
log = logging.getLogger('Task2Alignment')
```

And set up an output directory:

```
from wptrec.save import OutRepo
output = OutRepo('data/metric-tables')
```

```
from wptrec import metrics
from wptrec.dimension import sum_outer
```

C.2 Data and Helpers

Most data loading is outsourced to `MetricInputs`. First we save the data mode where metric inputs can find it:

```
import wptrec
wptrec.DATA_MODE = DATA_MODE
```

```
from MetricInputs import *
```

```
INFO:MetricInputs:reading data\metric-tables\page-sub-geo-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-src-geo-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-gender-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-occ-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-alpha-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-age-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-pop-align.parquet
INFO:MetricInputs:reading data\metric-tables\page-langs-align.parquet
```

```
dimensions
```

```
[<dimension "sub-geo": 21 levels>,
 <dimension "src-geo": 21 levels>,
 <dimension "gender": 4 levels>,
 <dimension "occ": 33 levels>,
 <dimension "alpha": 4 levels>,
 <dimension "age": 4 levels>,
 <dimension "pop": 4 levels>,
 <dimension "langs": 3 levels>]
```

C.2.1 qrel join

We want a function to join alignments with qrels:

```
def qr_join(align):
    return qrels.join(align, on='page_id').set_index(['topic_id', 'page_id'])
```

C.2.2 norm_dist

And a function to normalize to a distribution:

```
def norm_dist_df(mat):
    sums = mat.sum('columns')
    return mat.divide(sums, 'rows')
```

C.3 Work and Target Exposure

The first thing we need to do to prepare the metric is to compute the work-needed for each topic's pages, and use that to compute the target exposure for each (relevant) page in the topic.

This is because an ideal ranking orders relevant documents in decreasing order of work needed, followed by irrelevant documents. All relevant documents at a given work level should receive the same expected exposure.

First, look up the work for each query page ('query page work', or qpw):

```
qpw = qrels.join(page_quality, on='page_id')
qpw
```

```
         topic_id   page_id quality
0             187       682       B
1             187       954       C
2             187      1170       C
3             187      1315       B
4             187      1322       B
...           ...       ...     ...
2737607      2872  69877511    Stub
2737608      2872  69878912       C
2737609      2872  69879322   Start
2737610      2872  69881345    Stub
2737611      2872  69883661   Start

[2737612 rows x 3 columns]
```

And now use that to compute the number of documents at each work level:

```
qwork = qpw.groupby(['topic_id', 'quality'])['page_id'].count()
qwork
```

```
topic_id  quality
187       Stub       31076
          Start      20015
          C          11853
          B           4146
          GA          1479
                     ...
2872      Start      21769
          C           9480
          B           2627
          GA           806
          FA            69
Name: page_id, Length: 300, dtype: int64
```

Now we need to convert this into target exposure levels. This function will, given a series of counts for each work level, compute the expected exposure a page at that work level should receive.

```
def qw_tgt_exposure(qw_counts: pd.Series) -> pd.Series:
    if 'topic_id' == qw_counts.index.names[0]:
        qw_counts = qw_counts.reset_index(level='topic_id', drop=True)
    qwc = qw_counts.reindex(work_order, fill_value=0).astype('i4')
    tot = int(qwc.sum())
```

```
    da = metrics.discount(tot)
    qwp = qwc.shift(1, fill_value=0)
    qwc_s = qwc.cumsum()
    qwp_s = qwp.cumsum()
    res = pd.Series(
        [np.mean(da[s:e]) for (s, e) in zip(qwp_s, qwc_s)],
        index=qwc.index
    )
    return res
```

We'll then apply this to each topic, to determine the per-topic target exposures:

```
qw_pp_target = qwork.groupby('topic_id').apply(qw_tgt_exposure)
qw_pp_target.name = 'tgt_exposure'
qw_pp_target
```

```
C:\Users\michaelekstrand\scoop\apps\mambaforge\current\envs\wptrec\lib\site-packages\numpy\core\fromnume
  return _methods._mean(a, axis=axis, dtype=dtype,
C:\Users\michaelekstrand\scoop\apps\mambaforge\current\envs\wptrec\lib\site-packages\numpy\core\_methods
  ret = ret.dtype.type(ret / rcount)
```

```
topic_id  quality
187       Stub       0.075443
          Start      0.065321
          C          0.063307
          B          0.062546
          GA         0.062307
                       ...
2872      Start      0.062570
          C          0.061352
          B          0.060958
          GA         0.060853
          FA         0.060827
Name: tgt_exposure, Length: 300, dtype: float32
```

We can now merge the relevant document work categories with this exposure, to compute the target exposure for each relevant document:

```
qp_exp = qpw.join(qw_pp_target, on=['topic_id', 'quality'])
qp_exp = qp_exp.set_index(['topic_id', 'page_id'])['tgt_exposure']
qp_exp
```

```
topic_id  page_id
187       682         0.062546
          954         0.063307
          1170        0.063307
          1315        0.062546
          1322        0.062546
                        ...
2872      69877511    0.071035
          69878912    0.061352
          69879322    0.062570
          69881345    0.071035
          69883661    0.062570
Name: tgt_exposure, Length: 2737612, dtype: float32
```

C.4 Subject Geography

Subject geography targets the average of the relevant set alignments and the world population.

```
qr_sub_geo_align = qr_join(sub_geo_align)
qr_sub_geo_align
```

```
                    @UNKNOWN  Antarctica  Caribbean  Central America  \
topic_id page_id
187      682             1.0         0.0        0.0              0.0
         954             0.0         0.0        0.0              0.0
         1170            1.0         0.0        0.0              0.0
         1315            1.0         0.0        0.0              0.0
         1322            0.0         0.0        0.0              0.0
...                      ...         ...        ...              ...
2872     69877511        1.0         0.0        0.0              0.0
         69878912        1.0         0.0        0.0              0.0
         69879322        1.0         0.0        0.0              0.0
         69881345        0.0         0.0        0.0              0.0
         69883661        1.0         0.0        0.0              0.0

                    Central Asia  Eastern Africa  Eastern Asia  Eastern Europe  \
topic_id page_id
187      682                 0.0             0.0           0.0             0.0
         954                 0.0             0.0           0.0             0.0
         1170                0.0             0.0           0.0             0.0
         1315                0.0             0.0           0.0             0.0
         1322                0.0             0.0           0.0             0.0
...                          ...             ...           ...             ...
2872     69877511            0.0             0.0           0.0             0.0
         69878912            0.0             0.0           0.0             0.0
         69879322            0.0             0.0           0.0             0.0
         69881345            0.0             0.0           0.0             0.0
         69883661            0.0             0.0           0.0             0.0

                    Middle Africa  Northern Africa  ...  Northern Europe  \
topic_id page_id                                    ...
187      682                  0.0              0.0  ...              0.0
         954                  0.0              0.0  ...              0.0
         1170                 0.0              0.0  ...              0.0
         1315                 0.0              0.0  ...              0.0
         1322                 0.0              0.0  ...              0.0
...                           ...              ...  ...              ...
2872     69877511             0.0              0.0  ...              0.0
         69878912             0.0              0.0  ...              0.0
         69879322             0.0              0.0  ...              0.0
         69881345             0.0              0.0  ...              0.0
         69883661             0.0              0.0  ...              0.0

                    Oceania  South America  South-eastern Asia  \
topic_id page_id
187      682            0.0            0.0                 0.0
         954            0.0            0.0                 0.0
```

```
         1170          0.0           0.0                0.0
         1315          0.0           0.0                0.0
         1322          0.0           0.0                0.0
...                    ...           ...                ...
2872     69877511      0.0           0.0                0.0
         69878912      0.0           0.0                0.0
         69879322      0.0           0.0                0.0
         69881345      0.0           0.0                1.0
         69883661      0.0           0.0                0.0

                   Southern Africa  Southern Asia  Southern Europe  \
topic_id page_id
187      682                   0.0            0.0              0.0
         954                   0.0            0.0              0.0
         1170                  0.0            0.0              0.0
         1315                  0.0            0.0              0.0
         1322                  0.0            0.0              1.0
...                            ...            ...              ...
2872     69877511              0.0            0.0              0.0
         69878912              0.0            0.0              0.0
         69879322              0.0            0.0              0.0
         69881345              0.0            0.0              0.0
         69883661              0.0            0.0              0.0

                   Western Africa  Western Asia  Western Europe
topic_id page_id
187      682                  0.0           0.0             0.0
         954                  0.0           0.0             1.0
         1170                 0.0           0.0             0.0
         1315                 0.0           0.0             0.0
         1322                 0.0           0.0             0.0
...                           ...           ...             ...
2872     69877511             0.0           0.0             0.0
         69878912             0.0           0.0             0.0
         69879322             0.0           0.0             0.0
         69881345             0.0           0.0             0.0
         69883661             0.0           0.0             0.0

[2737612 rows x 21 columns]
```

Compute a raw target, factoring in weights:

```
qr_sub_geo_tgt = qr_sub_geo_align.multiply(qp_exp, axis='rows').groupby('topic_id').sum()
```

And now we need to average the known-geo with the background.

```
qr_sub_geo_fk = qr_sub_geo_tgt.iloc[:, 1:].sum('columns')
qr_sub_geo_tgt.iloc[:, 1:] *= 0.5
qr_sub_geo_tgt.iloc[:, 1:] += qr_sub_geo_fk.apply(lambda k: world_pop * k * 0.5)
qr_sub_geo_tgt.head()
```

```
             @UNKNOWN  Antarctica  Caribbean  Central America  Central Asia  \
topic_id
```

```
187        758.390795    0.000309  19.328449        59.318134     21.122305
270        967.129024    0.000233  69.231601        59.217295     23.466741
359        641.628435    0.000220  59.452325        50.316697     12.890808
365        481.821710    0.000148  18.649567        31.101461      9.853304
400       2137.392223    0.000465  39.636681       106.665337     28.905838

          Eastern Africa  Eastern Asia  Eastern Europe  Middle Africa  \
topic_id
187           109.070472    538.041050      160.059755      39.543253
270           147.703296    423.954414      216.488611      39.756533
359            74.364014    418.392361       59.464259      27.060083
365            53.778891    251.973232       88.152130      28.231344
400           168.787290    825.397926      224.909498      61.032641

          Northern Africa  ...  Northern Europe     Oceania  South America  \
topic_id                   ...
187             70.002691  ...       629.703684   93.268184     144.134466
270             68.587236  ...       234.564466   82.918686     151.424207
359             41.801031  ...        23.286746   19.606259     102.636117
365             36.627649  ...        70.381866   38.938076      88.112552
400            101.411871  ...       623.047574  189.240030     250.638365

          South-eastern Asia  Southern Africa  Southern Asia  Southern Europe  \
topic_id
187               206.460548        22.555736     554.706743       289.109630
270               154.497647        35.594573     402.112346       174.454966
359               124.679452        12.871306     348.961634        36.449902
365               128.244847         9.595809     242.239786       158.985835
400               297.782968        44.412480     820.697054       224.451157

          Western Africa  Western Asia  Western Europe
topic_id
187            97.058510    112.506820      279.935254
270           106.516209     91.341129      224.075828
359            66.136114     49.743391       42.552754
365            58.330269     70.821535       78.130276
400           154.170802    152.891150      466.963698

[5 rows x 21 columns]
```

These are **not** distributions, let's fix that!

```
qr_sub_geo_tgt = norm_dist_df(qr_sub_geo_tgt)
```

```
output.save_table(qr_sub_geo_tgt, f'task2-{DATA_MODE}-sub-geo-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task2-eval-sub-geo-target.csv.gz
INFO:wptrec.save:data\metric-tables\task2-eval-sub-geo-target.csv.gz: 10.67 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task2-eval-sub-geo-target.parquet
INFO:wptrec.save:data\metric-tables\task2-eval-sub-geo-target.parquet: 25.97 KiB
```

C.5 Source Geography

Source geography works the same way.

```
qr_src_geo_align = qr_join(src_geo_align)
qr_src_geo_align
```

```
                  @UNKNOWN  Antarctica  Caribbean  Central America  \
topic_id page_id
187      682      0.400000         0.0        0.0              0.0
         954      0.257143         0.0        0.0              0.0
         1170     0.368421         0.0        0.0              0.0
         1315     0.375000         0.0        0.0              0.0
         1322     0.428571         0.0        0.0              0.0
...                    ...         ...        ...              ...
2872     69877511 1.000000         0.0        0.0              0.0
         69878912 0.366667         0.0        0.0              0.0
         69879322 0.200000         0.0        0.0              0.0
         69881345 0.500000         0.0        0.0              0.0
         69883661 0.000000         0.0        0.0              0.0

                  Central Asia  Eastern Africa  Eastern Asia  Eastern Europe  \
topic_id page_id
187      682               0.0             0.0           0.0             0.0
         954               0.0             0.0           0.0             0.0
         1170              0.0             0.0           0.0             0.0
         1315              0.0             0.0           0.0             0.0
         1322              0.0             0.0           0.0             0.0
...                        ...             ...           ...             ...
2872     69877511          0.0             0.0           0.0             0.0
         69878912          0.0             0.0           0.0             0.0
         69879322          0.0             0.0           0.0             0.0
         69881345          0.0             0.0           0.0             0.0
         69883661          0.0             0.0           0.0             0.0

                  Middle Africa  Northern Africa  ...  Northern Europe  \
topic_id page_id                                  ...
187      682                0.0              0.0  ...         0.150000
         954                0.0              0.0  ...         0.285714
         1170               0.0              0.0  ...         0.052632
         1315               0.0              0.0  ...         0.000000
         1322               0.0              0.0  ...         0.000000
...                         ...              ...  ...              ...
2872     69877511           0.0              0.0  ...         0.000000
         69878912           0.0              0.0  ...         0.000000
         69879322           0.0              0.0  ...         0.000000
         69881345           0.0              0.0  ...         0.000000
         69883661           0.0              0.0  ...         0.000000

                   Oceania  South America  South-eastern Asia  \
topic_id page_id
187      682      0.000000            0.0                 0.0
         954      0.000000            0.0                 0.0
         1170     0.052632            0.0                 0.0
         1315     0.000000            0.0                 0.0
         1322     0.000000            0.0                 0.0
```

```
...                     ...             ...                  ...
2872      69877511  0.000000            0.0                  0.0
          69878912  0.000000            0.0                  0.1
          69879322  0.000000            0.0                  0.0
          69881345  0.000000            0.0                  0.5
          69883661  0.000000            0.0                  0.0

                   Southern Africa  Southern Asia  Southern Europe  \
topic_id page_id
187      682                   0.0            0.0         0.000000
         954                   0.0            0.0         0.000000
         1170                  0.0            0.0         0.000000
         1315                  0.0            0.0         0.000000
         1322                  0.0            0.0         0.571429
...                            ...            ...              ...
2872     69877511              0.0            0.0         0.000000
         69878912              0.0            0.0         0.000000
         69879322              0.0            0.0         0.000000
         69881345              0.0            0.0         0.000000
         69883661              0.0            0.0         0.000000

                   Western Africa  Western Asia  Western Europe
topic_id page_id
187      682                  0.0         0.000        0.050000
         954                  0.0         0.000        0.171429
         1170                 0.0         0.000        0.000000
         1315                 0.0         0.125        0.000000
         1322                 0.0         0.000        0.000000
...                           ...           ...             ...
2872     69877511             0.0         0.000        0.000000
         69878912             0.0         0.000        0.000000
         69879322             0.0         0.600        0.000000
         69881345             0.0         0.000        0.000000
         69883661             0.0         0.000        0.000000

[2737612 rows x 21 columns]
```

And now we repeat these computations!

```
qr_src_geo_tgt = qr_src_geo_align.multiply(qp_exp, axis='rows').groupby('topic_id').sum()
```

```
qr_src_geo_fk = qr_src_geo_tgt.iloc[:, 1:].sum('columns')
qr_src_geo_tgt.iloc[:, 1:] *= 0.5
qr_src_geo_tgt.iloc[:, 1:] += qr_src_geo_fk.apply(lambda k: world_pop * k * 0.5)
qr_src_geo_tgt.head()
```

```
             @UNKNOWN  Antarctica  Caribbean  Central America  Central Asia  \
topic_id
187       1892.369467    0.000221  10.629244        38.085204     13.580445
270       1682.383393    0.000177  14.119208        32.195247     10.694027
359       1349.305462    0.000166  10.812019        28.257371      9.637102
365        899.571884    0.000116  24.578317        20.163280      7.058418
400       3510.441727    0.002120  20.067844        67.028356     21.829953
```

```
          Eastern Africa  Eastern Asia  Eastern Europe  Middle Africa  \
topic_id
187            76.133518    368.324160       91.512896      27.292950
270            62.403539    291.840082       76.966103      21.889625
359            55.966959    288.368964       44.738826      20.314799
365            40.687112    195.637555       43.383442      18.290350
400           124.063329    603.794372      146.012407      44.342043

          Northern Africa  ...  Northern Europe     Oceania  South America  \
topic_id                   ...
187             43.321614  ...       518.410807   53.513703      92.059446
270             34.559267  ...       175.354880   40.800385      81.370613
359             31.459874  ...        27.303292   12.455743      62.663487
365             22.941681  ...        51.127647   29.052902      48.157030
400             71.306778  ...       564.339012  159.334304     158.261887

          South-eastern Asia  Southern Africa  Southern Asia  Southern Europe  \
topic_id
187               137.914296        14.561724     383.654770        94.527498
270               104.008024        13.928225     291.129055        74.411532
359                93.284401         9.119594     262.223930        24.364869
365                91.720399         6.467410     185.621213        93.737043
400               220.074822        27.370868     631.930727       129.760455

          Western Africa  Western Asia  Western Europe
topic_id
187            67.295433     64.363892      158.997834
270            54.628367     45.613038      244.158227
359            49.612454     37.221849       34.950278
365            35.679642     47.403614       70.055359
400           111.504376    103.379965      207.421067

[5 rows x 21 columns]
```

Make sure the rows are distributions:

```
qr_src_geo_tgt = norm_dist_df(qr_src_geo_tgt)
```

```
output.save_table(qr_src_geo_tgt, f'task2-{DATA_MODE}-src-geo-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task2-eval-src-geo-target.csv.gz
INFO:wptrec.save:data\metric-tables\task2-eval-src-geo-target.csv.gz: 10.62 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task2-eval-src-geo-target.parquet
INFO:wptrec.save:data\metric-tables\task2-eval-src-geo-target.parquet: 25.97 KiB
```

C.6 Gender

Now we're going to grab the gender alignments. Works the same way.

```
qr_gender_align = qr_join(gender_align)
qr_gender_align.head()
```

```
                  @UNKNOWN  female  male   NB
topic_id page_id
187      682           1.0     0.0   0.0  0.0
         954           0.0     0.0   1.0  0.0
         1170          1.0     0.0   0.0  0.0
         1315          1.0     0.0   0.0  0.0
         1322          1.0     0.0   0.0  0.0
```

```
qr_gender_tgt = qr_gender_align.multiply(qp_exp, axis='rows').groupby('topic_id').sum()
```

```
qr_gender_fk = qr_gender_tgt.iloc[:, 1:].sum('columns')
qr_gender_tgt.iloc[:, 1:] *= 0.5
qr_gender_tgt.iloc[:, 1:] += qr_gender_fk.apply(lambda k: gender_tgt * k * 0.5)
qr_gender_tgt.head()
```

```
             @UNKNOWN       female         male         NB
topic_id
187       4231.726279   159.708759   364.436851   2.704633
270       1461.677295  1029.567013  1476.707985  12.917147
359       1164.868940   601.468537  1714.967051  11.640380
365       1012.069178   445.784544   938.953553   6.958483
400         94.885554  3323.222661  4707.223206  42.888097
```

```
qr_gender_tgt = norm_dist_df(qr_gender_tgt)
```

```
output.save_table(qr_gender_tgt, f'task2-{DATA_MODE}-gender-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task2-eval-gender-target.csv.gz
INFO:wptrec.save:data\metric-tables\task2-eval-gender-target.csv.gz: 2.24 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task2-eval-gender-target.parquet
INFO:wptrec.save:data\metric-tables\task2-eval-gender-target.parquet: 6.90 KiB
```

C.7 Occupation

Occupation is more straightforward, since we don't have a global target to average with. We do need to drop unknown.

```
qr_occ_align = qr_join(occ_align).multiply(qp_exp, axis='rows')
qr_occ_tgt = qr_occ_align.iloc[:, 1:].groupby('topic_id').sum()
qr_occ_tgt = norm_dist_df(qr_occ_tgt)
qr_occ_tgt.head()
```

```
          activist  agricultural worker    artist   athlete  biologist  \
topic_id
187       0.001742             0.000423  0.046779  0.003448   0.001719
270       0.000220             0.000236  0.000887  0.963747   0.000225
359       0.000308             0.000073  0.000855  0.913443   0.000066
365       0.000134             0.000030  0.000319  0.874820   0.000039
400       0.004460             0.000402  0.331925  0.003775   0.001594

          businessperson   chemist  civil servant  clergyperson  \
topic_id
187             0.025363  0.000046       0.001743      0.000760
```

```
270            0.001724  0.000190       0.001045      0.000168
359            0.007293  0.000094       0.000495      0.000071
365            0.003116  0.000089       0.000365      0.000151
400            0.020481  0.000277       0.002385      0.001815

          computer scientist  ...  military personnel  musician  \
topic_id                      ...
187                 0.000127  ...            0.003020  0.001195
270                 0.000015  ...            0.001331  0.000621
359                 0.000000  ...            0.002207  0.001371
365                 0.000024  ...            0.001284  0.000451
400                 0.000278  ...            0.002132  0.011309

          performing artist  physicist  politician  scientist  \
topic_id
187                0.000999   0.000467    0.009501   0.010534
270                0.001608   0.000035    0.003671   0.000428
359                0.004169   0.000014    0.002663   0.000054
365                0.002861   0.000000    0.001718   0.000100
400                0.133634   0.000404    0.007652   0.003079

          social scientist  sportsperson (non-athlete)  \
topic_id
187               0.004196                    0.000352
270               0.000431                    0.013288
359               0.000069                    0.047321
365               0.000131                    0.106472
400               0.003482                    0.001700

          transportation occupation    writer
topic_id
187                        0.000268  0.012910
270                        0.000434  0.001255
359                        0.000085  0.002104
365                        0.000160  0.001281
400                        0.000531  0.262259

[5 rows x 32 columns]
```

```python
output.save_table(qr_occ_tgt, f'task2-{DATA_MODE}-occ-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task2-eval-occ-target.csv.gz
INFO:wptrec.save:data\metric-tables\task2-eval-occ-target.csv.gz: 14.69 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task2-eval-occ-target.parquet
INFO:wptrec.save:data\metric-tables\task2-eval-occ-target.parquet: 37.48 KiB
```

C.8 Remaining Attributes

The remaining attributes don't need any further processing, as they are completely known.

```python
qr_age_align = qr_join(age_align).multiply(qp_exp, axis='rows')
qr_age_tgt = norm_dist_df(qr_age_align.groupby('topic_id').sum())
output.save_table(qr_age_tgt, f'task2-{DATA_MODE}-age-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task2-eval-age-target.csv.gz
INFO:wptrec.save:data\metric-tables\task2-eval-age-target.csv.gz: 1.20 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task2-eval-age-target.parquet
INFO:wptrec.save:data\metric-tables\task2-eval-age-target.parquet: 5.24 KiB
```

```
qr_alpha_align = qr_join(alpha_align).multiply(qp_exp, axis='rows')
qr_alpha_tgt = norm_dist_df(qr_alpha_align.groupby('topic_id').sum())
output.save_table(qr_alpha_tgt, f'task2-{DATA_MODE}-alpha-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task2-eval-alpha-target.csv.gz
INFO:wptrec.save:data\metric-tables\task2-eval-alpha-target.csv.gz: 1.16 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task2-eval-alpha-target.parquet
INFO:wptrec.save:data\metric-tables\task2-eval-alpha-target.parquet: 5.00 KiB
```

```
qr_langs_align = qr_join(langs_align).multiply(qp_exp, axis='rows')
qr_langs_tgt = norm_dist_df(qr_langs_align.groupby('topic_id').sum())
output.save_table(qr_langs_tgt, f'task2-{DATA_MODE}-langs-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task2-eval-langs-target.csv.gz
INFO:wptrec.save:data\metric-tables\task2-eval-langs-target.csv.gz: 978.00 iB
INFO:wptrec.save:saving Parquet to data\metric-tables\task2-eval-langs-target.parquet
INFO:wptrec.save:data\metric-tables\task2-eval-langs-target.parquet: 4.46 KiB
```

```
qr_pop_align = qr_join(pop_align).multiply(qp_exp, axis='rows')
qr_pop_tgt = norm_dist_df(qr_pop_align.groupby('topic_id').sum())
output.save_table(qr_pop_tgt, f'task2-{DATA_MODE}-pop-target', parquet=True)
```

```
INFO:wptrec.save:saving CSV to data\metric-tables\task2-eval-pop-target.csv.gz
INFO:wptrec.save:data\metric-tables\task2-eval-pop-target.csv.gz: 1.24 KiB
INFO:wptrec.save:saving Parquet to data\metric-tables\task2-eval-pop-target.parquet
INFO:wptrec.save:data\metric-tables\task2-eval-pop-target.parquet: 5.15 KiB
```

C.9 Multidimensional Alignment

Now let's dive into the multidmensional alignment. This is going to proceed a lot like the Task 1 alignment.

C.9.1 Dimension Definitions

Let's define background distributions for some of our dimensions:

```
dim_backgrounds = {
    'sub-geo': world_pop,
    'src-geo': world_pop,
    'gender': gender_tgt,
}
```

Now we'll make a list of dimensions to treat with averaging:

```
DR = namedtuple('DimRec', ['name', 'align', 'background'], defaults=[None])
avg_dims = [
    DR(d.name, d.page_align_xr, xr.DataArray(dim_backgrounds[d.name], dims=[d.name]))
    for d in dimensions
    if d.name in dim_backgrounds
]
[d.name for d in avg_dims]
```

```
['sub-geo', 'src-geo', 'gender']
```

And a list of dimensions to use as-is:

```
raw_dims = [
    DR(d.name, d.page_align_xr)
    for d in dimensions
    if d.name not in dim_backgrounds
]
[d.name for d in raw_dims]
```

```
['occ', 'alpha', 'age', 'pop', 'langs']
```

Now: these dimension are in the original order - `dimensions` has the averaged dimensions before the non-averaged ones. **This is critical for the rest of the code to work.**

C.9.2 Data Subsetting

Also from Task 1.

```
avg_cases = list(product(*[[True, False] for d in avg_dims]))
avg_cases.pop()
avg_cases
```

```
[(True, True, True),
 (True, True, False),
 (True, False, True),
 (True, False, False),
 (False, True, True),
 (False, True, False),
 (False, False, True)]
```

```
def case_selector(case):
    def mksel(known):
        if known:
            # select all but 1st column
            return slice(1, None, None)
        else:
            # select 1st column
            return 0

    return tuple(mksel(k) for k in case)
```

C.9.3 Background Averaging

We're now going to define our background-averaging function; this is reused from the Task 1 alignment code.

For each condition, we are going to proceed as follows:

1. Compute an appropriate intersectional background distribution (based on the dimensions that are "known")
2. Select the subset of the target matrix with this known status
3. Compute the sum of this subset
4. Re-normalize the subset to sum to 1

5. Compute a normalization table such that each coordinate in the distributions to correct sums to 1 (so multiplying this by the background distribution spreads the background across the other dimensions appropriately), and use this to spread the background distribution
6. Average with the spread background distribution
7. Re-normalize to preserve the original sum

Let's define the whole process as a function:

```
def avg_with_bg(tm, verbose=False):
    tm = tm.copy()

    tail_names = [d.name for d in raw_dims]

    # compute the tail mass for each coordinate (can be done once)
    tail_mass = tm.sum(tail_names)

    # now some things don't have any mass, but we still need to distribute background distributions.
    # solution: we impute the marginal tail distribution
    # first compute it
    tail_marg = tm.sum([d.name for d in avg_dims])
    # then impute that where we don't have mass
    tm_imputed = xr.where(tail_mass > 0, tm, tail_marg)
    # and re-compute the tail mass
    tail_mass = tm_imputed.sum(tail_names)
    # and finally we compute the rescaled matrix
    tail_scale = tm_imputed / tail_mass
    del tm_imputed

    for case in avg_cases:
        # for deugging: get names
        known_names = [d.name for (d, known) in zip(avg_dims, case) if known]
        if verbose:
            print('processing known:', known_names)

        # Step 1: background
        bg = reduce(operator.mul, [
            d.background
            for (d, known) in zip(avg_dims, case)
            if known
        ])
        if not np.allclose(bg.sum(), 1.0):
            warnings.warn('background distribution for {} sums to {}, expected 1'.format(known_names, bg

        # Step 2: selector
        sel = case_selector(case)

        # Steps 3: sum in preparation for normalization
        c_sum = tm[sel].sum()

        # Step 5: spread the background
        bg_spread = bg * tail_scale[sel] * c_sum
        if not np.allclose(bg_spread.sum(), c_sum):
```

```
            warnings.warn('rescaled background sums to {}, expected c_sum'.format(bg_spread.values.sum()

        # Step 4 & 6: average with the background
        tm[sel] *= 0.5
        bg_spread *= 0.5
        tm[sel] += bg_spread

        if not np.allclose(tm[sel].sum(), c_sum):
            warnings.warn('target distribution for {} sums to {}, expected {}'.format(known_names, tm[se

    return tm
```

C.9.4 Computing Targets

We're now ready to compute a multidimensional target. This works like the Task 1, with the difference that we are propagating work needed into the targets as well; the input will be series whose *index* is page IDs and values are the work levels.

```
def query_xalign(pages):
    # compute targets to average
    avg_pages = reduce(operator.mul, [d.align.loc[pages.index] for d in avg_dims])
    raw_pages = reduce(operator.mul, [d.align.loc[pages.index] for d in raw_dims])

    # weight the left pages
    pages.index.name = 'page'
    qpw = xr.DataArray.from_series(pages)
    avg_pages = avg_pages * qpw

    # convert to query distribution
    tgt = sum_outer(avg_pages, raw_pages)
    tgt /= qpw.sum()

    # average with background distributions
    tgt = avg_with_bg(tgt)

    # and return the result
    return tgt
```

C.9.5 Applying Computations

Now let's run this thing - compute all the target distributions:

```
q_ids = qp_exp.index.levels[0].copy()
q_ids
```

```
Int64Index([ 187,  270,  359,  365,  400,  404,  480,  517,  568,  596,  715,
             807,  834,  881,  883,  949,  951,  955,  995, 1018, 1180, 1233,
            1328, 1406, 1417, 1448, 1449, 1479, 1499, 1548, 1558, 1647, 1685,
            1806, 1821, 1877, 1884, 1890, 2000, 2028, 2106, 2153, 2160, 2229,
            2244, 2448, 2483, 2758, 2867, 2872],
           dtype='int64', name='topic_id')
```

```
q_tgts = [query_xalign(qp_exp.loc[q]) for q in tqdm(q_ids)]
```

{"model_id":"825dff5cd101402e8910af2cb8a4abf7","version_major":2,"version_minor":0}

```
q_tgts = xr.concat(q_tgts, q_ids)
q_tgts
```

```
<xarray.DataArray (topic_id: 50, sub-geo: 21, src-geo: 21, gender: 4, occ: 33,
                   alpha: 4, age: 4, pop: 4, langs: 3)>
array([[[[[[[[[5.32222994e-10, 1.22700201e-08, 0.00000000e+00],
              [1.62526437e-08, 1.59783339e-08, 1.29447847e-08],
              [3.42041043e-09, 1.26698684e-08, 7.92859105e-10],
              [5.93547427e-09, 5.09430447e-09, 2.95050019e-09]],

             [[1.14269430e-10, 6.14178489e-10, 3.43674300e-11],
              [7.37725527e-09, 4.95719395e-09, 1.18193819e-08],
              [1.13716762e-09, 2.68869541e-09, 2.74025688e-10],
              [4.25413257e-09, 3.79956877e-09, 4.77083603e-09]],

             [[2.66154076e-11, 1.00500571e-10, 0.00000000e+00],
              [5.05224537e-09, 2.29518724e-09, 7.76356299e-09],
              [6.48676034e-10, 3.73767879e-10, 2.92756686e-10],
              [1.25282632e-09, 1.10224187e-09, 2.08692884e-09]],

             [[3.86682637e-10, 5.77107369e-11, 9.71290153e-11],
              [3.89256821e-09, 1.41817503e-09, 1.16465576e-08],
              [1.19974875e-10, 2.22190558e-11, 6.51717199e-10],
              [1.69738191e-09, 2.74512105e-10, 1.28763261e-09]]],

...

            [[[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

             [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

             [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]],

             [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
              [0.00000000e+00, 0.00000000e+00, 0.00000000e+00]]]]]]]]])
Coordinates:
  * sub-geo    (sub-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
  * src-geo    (src-geo) object '@UNKNOWN' 'Antarctica' ... 'Western Europe'
  * gender     (gender) object '@UNKNOWN' 'female' 'male' 'NB'
```

```
  * occ        (occ) object '@UNKNOWN' 'activist' ... 'writer'
  * alpha      (alpha) object 'a-d' 'e-k' 'l-r' 's-'
  * age        (age) object '2001-2006' '2007-2011' '2012-2016' '2017-2022'
  * pop        (pop) object 'High' 'Low' 'Medium-High' 'Medium-Low'
  * langs      (langs) object '2-4 languages' '5+ languages' 'English only'
  * topic_id   (topic_id) int64 187 270 359 365 400 ... 2448 2483 2758 2867 2872
```

Save this to NetCDF (xarray's recommended format):

```
output.save_xarray(q_tgts, f'task2-{DATA_MODE}-int-targets')
```

```
INFO:wptrec.save:saving NetCDF to data\metric-tables\task2-eval-int-targets.nc
```

C.10 Task 2B - Equity of Underexposure - NOT YET DONE

For 2022, we are using a diffrent version of the metric. **Equity of Underexposure** looks at each page's underexposure (system exposure is less than target exposure), and looks for underexposure to be equitably distributed between groups.

On its own, this isn't too difficult; averaging with background distributions, however, gets rather subtle. Background distributions are at the roup level, but we need to propgagate that into the page level, so we can compute the difference between system and target exposure at the page level, and then aggregate the underexposure within each group.

The idea of equity of underexposure is that we $\epsilon = \mathrm{E}_\pi[\eta]$ and $\epsilon^* = \mathrm{E}_\tau[\eta]$. We then compute $u = min(\epsilon^* - \epsilon, 0)$, and restrict it to be negative, and aggregate it by group; if A is our page alignment matrix and $\vec{u}$, we compute the group underexposure by $A^T\vec{u}$.

That's the key idea. However, we want to use $\epsilon^\dagger$ that has the equivalent of averaging group-aggregated ϵ^* with global target distributions w_g. We can do this in a few stages. First, we compute the total attention of each group, and use that to compute the fraction of group global weight that should go to each unit of alignment:

\begin{align*} s_g & = \sum_d a_{dg} \ \hat{w}_g & = \frac{w_g}{s_g} \end{align*}

We can then average:

\begin{align} *\epsilon^\dagger_d & = \frac{1}{2}\left(\epsilon*^_d + \sum_g a_{dg} \hat{w}_g \epsilon^*_{\mathrm{tot
\right) \ \end{align*}

This is all on a per-topic basis.

C.10.1 Demo Topic

We're going to reuse demo topic data from before:

```
q_xa
```

Compute the total for each attribute:

```
s_xg = q_xa.sum(axis=0) + 1e-10
s_xg
```

Let's get some fractions out of that:

```
s_xgf = s_xg / s_xg.sum()
s_xgf
```

Now, let's make a copy, and start building up a world target matrix that properly accounts for missing values:

```
W = s_xgf.copy()
```

Now, let's put in the known intersectional targets:

```
W[1:, 1:] = int_tgt * W[1:, 1:].sum()
```

Now we need the known-gender / unknown-geo targets:

```
W[0, 1:] = int_tgt.sum(axis=0) * W[0, 1:].sum()
```

And the known-geo / unknown-gender targets:

```
W[1:, 0] = int_tgt.sum(axis=1) * W[1:, 0].sum()
```

Let's see what we have:

```
W
```

Now we normalize it by s_g:

```
Wh = W / s_xg
Wh
```

The massive values are only where we have no relevant items, so they'll never actually be used. We can now compute the query-aligned target matrix.

```
qp_gt = (q_xa * (Wh * qp_exp[1].sum())).sum(axis=(1,2)).to_series()
qp_gt.index.name = 'page_id'
qp_gt
```

```
qp_exp[1]
```

```
qp_tgt = 0.5 * (qp_exp[1] + qp_gt)
qp_tgt
```

C.10.2 Setting Up Matrix

Now that we have the math worked out, we can create actual global target frames for each query.

```
def topic_page_tgt(qdf):
    pages = qdf['page_id']
    pages = pages[pages.isin(page_xalign.indexes['page'])]
    q_xa = page_xalign.loc[pages.values, :, :]

    # now we need to get the exposure for the pages
    p_exp = qp_exp.loc[qdf.name]
    assert p_exp.index.is_unique

    # need our sums
    s_xg = q_xa.sum(axis=0) + 1e-10

    # set up the global target
    W = s_xg / s_xg.sum()
    W[1:, 1:] = int_tgt * W[1:, 1:].sum()
    W[0, 1:] = int_tgt.sum(axis=0) * W[0, 1:].sum()
```

```python
    W[1:, 0] = int_tgt.sum(axis=1) * W[1:, 0].sum()

    # per-unit global weights, de-normalized by total exposure
    Wh = W / s_xg
    Wh *= p_exp.sum()

    # compute global target
    gtgt = q_xa * Wh
    gtgt = gtgt.sum(axis=(1,2)).to_series()

    # compute average target
    avg_tgt = 0.5 * (p_exp + gtgt)
    avg_tgt.index.name = 'page'

    return avg_tgt
```

Test it quick:

```python
topic_page_tgt(qdf)
```

And create our targets:

```python
qp_tgt = qrels.groupby('id').progress_apply(topic_page_tgt)
qp_tgt

save_table(qp_tgt.to_frame('target'), 'task2-all-page-targets')

train_qptgt = qp_tgt.loc[train_topics['id']].to_frame('target')
eval_qptgt = qp_tgt.loc[eval_topics['id']].to_frame('target')

save_table(train_qptgt, 'task2-train-page-targets')
save_table(eval_qptgt, 'task2-eval-page-targets')
```