Blending Vertical and Web results

A Case Study using Video Intent

Damien Lefortier[1,2], Pavel Serdyukov[1], Fedor Romanenko[1], and Maarten de Rijke[2]

[1] Yandex, Moscow {damien, pavser, fedor}@yandex-team.ru
[2] ISLA, University of Amsterdam, derijke@uva.nl

Abstract. Modern search engines aggregate results from specialized verticals into the Web search results. We study a setting where vertical and Web results are *blended* into a single result list, a setting that has not been studied before. We focus on video intent and present a detailed observational study of Yandex's two video content sources (i.e., the specialized vertical and a subset of the general web index) thus providing insights into their complementary character. By investigating how to blend results from these sources, we contrast traditional federated search and fusion-based approaches with newly proposed approaches that significantly outperform the baseline methods.

1 Introduction

Modern search engines integrate results from specialized search services, or verticals, into their Web search results. This task is called aggregated search [11] and is usually decomposed into *vertical selection* [1] and *result merging* [3, 13]. Aggregated search aims at diversifying the search engine result page (SERP) in order to provide results for different possible information needs, or intents, in different types of content (e.g., images, videos). Previous work often assumes a so-called block search result presentation, where vertical results are grouped together in a block and placed amongst the generic web results; see Fig. 1 (left). Instead, we assume a search result presentation in which homogeneous vertical and Web results are *blended* into a single result list, as in Fig. 1 (right), *before* being placed amongst other results, a setting that has not been studied before. In this setting, vertical and Web results blended together should be homogeneous because incoherent result presentation can result in user dissatisfaction [5].

We focus on video intent, i.e., the user intent that is satisfied by videos (in the same way that image intent is satisfied by images). Video intent is broad and diverse with queries such as "*games of thrones watch video*" or "*the great gatsby.*" To answer such queries commercial search engines have two content sources: (1) A specialized video vertical that typically only contains videos from video hosting services, such as Youtube; we refer to this vertical as the *Video* vertical. (2) The subset of pages from the general web index with at least one video embedded in them; one can easily identify such pages and we refer to the set of such pages as the *WebVideo* source. These two sources correspond to different products and have different underlying ranking systems.

Is one of these two sources sufficient? To what extent do the Video and WebVideo sources complement each other as sources for answering the video intent in terms of user satisfaction? We investigate these questions through a case study of Yandex's two video content sources. We examine the hypothesis that the two sources satisfy different video information needs and complement each other in this respect, i.e., that both sources are required in order to more effectively satisfy information needs with a video intent. One striking example of such a need (query) is "*games of thrones watch video*," for which a user might want to watch the latest episode of the series (such videos are usually ranked sufficiently high only in Video) or to browse through all previous episodes to watch some of them (such hub pages are usually included only in WebVideo): these information needs are actually sub-intents of the video intent. These information needs are both satisfied by watching a video online, but the videos are typically found in different types of source (Video and Webvideo) so that both types of source are required to address queries with a video intent.

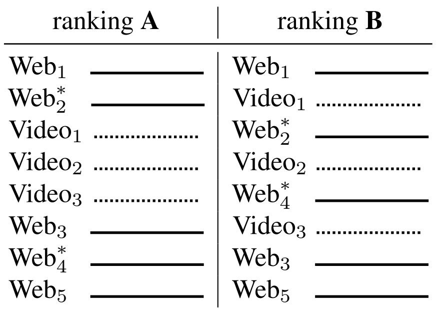

Fig. 1: Traditional block (left) vs. blended (right) search result presentations, where video vertical documents are shown as dotted lines, while pages from the general web index that have at least one video embedded in them are marked with *.

Below, we start with an observational comparison of the two types of video content source that we consider. We find that they complement each other and that both are required to best answer video information needs. For many queries, the two sources return highly relevant, but different results and, in such cases, presenting results from both sources is better for diversity than using a single source. In some cases, one of the two sources does not provide highly relevant results even for queries with a strong, dominant video intent. Therefore, by itself query intent is not enough to properly decide when to present results from a particular source, and the relevance of the results w.r.t. sub-intents of the underlying video intent must be taken into account. I.e., we also need to consider the ability of a particular source to answer some classes of queries.

These observations motivate us to address the following algorithmic problem: which results from the two sources should be presented on the SERP? Specifically, we consider the following *blending problem*: given the result lists from the two sources, produce the best possible result list in terms of overall relevance, where documents from each source can be interleaved (blended) together, subject to the constraint that no change is allowed in the relative order of results from the same source (see below).

The main challenge with our blending problem lies in the fact that document scores are not comparable across the two sources, because they have completely different underlying ranking systems. Our blending problem is similar to an aggregated or federated search problem, but it differs in the following essential ways. First, we blend documents and not blocks, as has been done so far in aggregated search [3, 13]. Second, compared to a federated search setting we exploit additional knowledge about each source (e.g., document scores returned by each source plus relevance assessments for training our models). Third, we do not want to change the relative order of results from the same source; we assume that the base rankers underlying the sources are the best experts about their own documents, so we do not overrule them [3, 16]; we refer to this constraint as the *no re-ordering constraint*. According to our experiments, our methods do not improve by relaxing this constraint; doing so only slows down the computation.

We consider a number of methods from federated search and data fusion. Their performance is found to be far below what could be achieved given the quality of the sources being blended. We therefore consider two types of alternative method. One predicts the quality of the top results from each source for a given query in order to blend results, but without directly using the document scores. The second is based on learning to rank (LTR); we propose three approaches: point-wise, pair-wise and list-wise. Our most effective blending approach belongs to the first type; it yields a significant +13.20% improvement in ERR@5 scores compared to using the best vertical only.

The main contributions of this paper are the following. (1) We present a detailed observational study of two video content sources thus providing insights into their complementary character. (2) We propose and evaluate different methods for blending results from these two sources into a single result list.

2 Related work

There are several tasks where documents are retrieved from multiple systems and merged into a single SERP. We review them and explain how our blending problem differs.

Aggregated Search. Prior work on aggregated search and result merging [3, 13] assumes a block search result presentation, so that, in our case, results from the Video vertical would be grouped together in a block and placed amongst the generic web results; see Fig. 1 (left). We, instead, assume a search result presentation in which vertical and Web results are blended into a single result list (right), a setting that has not been studied before. Aggregated search methods are not usable to solve our problem, because we blend documents and not blocks, as is customary in aggregated search [3, 13]

Federated Search. In federated search [16], of which aggregated search is a possible application, documents can come from several search engines. Popular result merging methods, such as CORI [4], use a combination of local collection scores and document scores to compute a global score for each document that is then used to rank results globally. We use variations of CORI as baselines (see §3.1), but not the original CORI method because it relies on term usage statistics from each collection. Such statistics are meaningful only when collections have documents of the same nature, which is not the case here as WebVideo indexes web pages, while Video indexes video metadata [18], which implies the use of a specialized ranking system designed for the video retrieval task. Other popular methods (e.g., SAFE [17]) rely on a centralized index and are not applicable to our task because such an index is unavailable to us.

Data Fusion. In many data fusion methods, results in the lists being fused come from a *single* collection, but are retrieved using different strategies. These methods, either unsupervised (e.g., CombSUM [6]) or supervised (e.g., λ-Merge [15], where result lists are retrieved using the same ranker unlike in our setting), are based on a voting principle and therefore some intersection between results is assumed. In our case, the intersection between the two video content sources is really small (see §6.1). Hence, fusion methods are not readily applicable to our blending problem. Still, we use some merging algorithms from data fusion as some of our baselines (see §3.1).

Learning to Rank. Learning to rank (LTR) approaches are not straightforward to use here because of the different feature spaces (one per source), which must be merged somehow. This was extensively discussed in [3], where different ways to construct feature vectors were proposed and compared in the context of block-ranking (ordering blocks of Web and vertical results in response to a query). In [3], the variant performing the best is one that makes a copy of each feature for each vertical, thus allowing the LTR model to learn a vertical-specific relationship between features and relevance. In this paper, we follow Arguello et al. [3] and also use *one copy* of each source feature for every source, which yields better performance according to our experiments as well. Then, to solve our problem of blending results, we propose a pair-wise approach inspired by [3, 15], but adapted to our problem as well as a point-wise approach similar in aim to the method described in [19], where it was outperformed by *Round-Robin* [21], and a list-wise approach that is the most fitted to our problem and allows us to use features from [12], where query performance was predicted.

In sum, we contribute a new problem (our blending problem) for which existing federated search and fusion methods are ill-fitted, plus algorithms to address the problem.

3 Blending methods

In this section, we describe different methods to address our blending problem. Recall that we do not change the relative order of documents that come from the same source. For example: W_1,W_2,V_1,V_2 is an allowed blending, while V_1,W_3,V_2,W_2 is disallowed as W_2 and W_3 were re-ordered, where V (W) means Video (WebVideo) and S_i is the i^{th} result from S. Given this constraint, there are, for a blended list of N results, 2^N possible blendings of the two sources.

3.1 Baselines We use the following baselines: *Round-Robin* [21], *Raw Score Merging* [10], and several variations of CORI merging [4]. The CORI result merging method computes a *global score* for each document from each source using a linear combination of the collection score C and the original document score D returned by each source:

$$\text{global score}(D) = \frac{D' + 0.4 \cdot D' \cdot C'}{1.4}$$

where C' (respectively D') is C (respectively D) MinMax-normalized to $[0, 1]$. The global score of each document is then used to rank results globally. We cannot use the CORI collection selection algorithm [4] to compute C (see §2). Instead we propose two variations, where D is the original document score and C is defined as follows:

CORI-Size. We define C to be the number of documents retrieved from each source (this was already proposed in [14]).

CORI-ML. We define C to be the predicted quality of each source as defined in §3.2, which makes this baseline our contribution as well.

3.2 Approaches based on source quality These methods consist of two steps: (1) we predict the quality of the top results from each source for a given query in order to (2) blend results from each source using this predicted quality (as in *CORI-ML*). Such methods are common in federated search (see §2) and the main problem lies in the uncertainty of how this predicted quality best translates into a weight for estimating the global score of each document. We therefore propose methods that do not use the original document scores returned by each source directly to overcome such issues (as opposed to baselines described in §3.1). Note that our methods are not suitable for a federated search setting as they are not readily applicable to more than two sources. Let $q_S(\overline{v}_q, \theta)$ be the *source quality function* of a source S with parameters θ, and where $\overline{v}_q$ is the vector of source features (see §4.1) for a query q. Using q_{Video} and q_{WebVideo}, we define the following blending methods:

Source-Binary. For each query, we show results from one source only, i.e., from Video if $q_{\text{Video}} > q_{\text{WebVideo}}$, and otherwise from WebVideo.

Source-KMeans. Let $x = q_{\text{Video}} - q_{\text{WebVideo}}$ be the predicted difference in quality of the two sources for a given query. For each possible difference x, the best uniform blending (i.e., the same blending for all queries) is found out of all possible blendings satisfying the *no re-ordering constraint*. E.g., for $x = 0.01$, it is V_1, W_1, V_2, W_2, while for $x = 0.6$, it is V_1,V_2,V_3,V_4. To find this blending, for a difference x, we take the k queries in the training data whose predicted difference in quality of the two sources is closest to x. For each query, we compute the score of each possible blending according to the retrieval metric used to train q_S. We then take the uniform blending with the highest average performance among these k queries. Here, k is a parameter that is determined experimentally on the training data.

We also investigated methods based on clustering the predicted values of quality (e.g., 0–0.20, 0.20–0.45, 0.45–1.0) and on defining a uniform blending depending on the class of each source. However, such methods did not yield better results than the *Source-KMeans* method and are omitted for brevity. We train q_S to predict ERR@N (where N is defined for the retrieval metric being used) by minimizing the mean squared error of the difference between the predicted and actual value of ERR for labeled result lists in the training data using Gradient Boosted Regression Trees (GBRT) [7]. Using ERR here yields better results than NDCG, both for ERR and NDCG.

3.3 Learning to rank approaches

Point-wise. The first of the three LTR approaches that we describe is a *point-wise* approach. In other words, we predict the *global score* of each document from each source that is then used to rank results globally using one independent model per source. Using one unified model does not improve the performance according to preliminary experiments on the training data, and is less scalable as the number of sources increases. Let $f_S(\overline{v}_q, \overline{w}_{qd}, \theta)$ be the *scoring function* of documents from a source S with parameters θ, where $\overline{v}_q$ and $\overline{w}_{qd}$ are, respectively, the vector of source features (see §4.2) and the vector of document features (see §4.3) for a query q and document d. This function predicts the global score of each document from source S. Using f_{Video} and f_{WebVideo}, we can easily define a blending algorithm, where the blended result list is the one with the highest value of the retrieval metric being used according to predicted scores; see Algorithm 1. Due to the *no re-ordering constraint*, we cannot simply rank documents by their predicted score. We train f_S to predict the score of each document obtained through judgements (see §5.2) by minimizing the mean squared error using GBRT [7].

Pair-wise. For this method, we use a *pair-wise* approach to LTR. In other words, we predict the *preference* between any pair of documents from Video *and* WebVideo in a unified model. Let $f(\overline{v}_q, \overline{w}_{qd}, \theta)$ be a *document preference function* with parameters θ,

Algorithm 1: Point-wise LTR blending algorithm

```
input  : Result list from $S$, $f_S$, $\theta_S$ for $S \in \{\text{Video}, \text{WebVideo}\}$, $\overline{v}_q$, $\overline{w}_{qd}$, $N$
output: Blended result list

BlendedResultList $\longleftarrow \{\}$; MaxScore $\longleftarrow 0$
for tuple $\in$ NAryCartesianProduct($\{0,1\}$) do
    Blending $\longleftarrow \{\}$ i $\longleftarrow 0$ j $\longleftarrow 0$
    for $x \in$ tuple do
        if $x == 0$ then
            Blending[i + j] = ResultList$_{\text{WebVideo}}$[i] // $i^{th}$ result from WebVideo
            i $\longleftarrow$ i + 1
        else
            Blending[i + j] = ResultList$_{\text{Video}}$[j]    // $j^{th}$ result from Video
            j $\longleftarrow$ j + 1
    Score $\longleftarrow$ Metric@N(Blending, $f_{\text{Video}}, \theta_{\text{Video}}, f_{\text{WebVideo}}, \theta_{\text{WebVideo}}, \overline{v}_q, \overline{w}_{qd}$)
    if Score > MaxScore then
        BlendedResultList $\longleftarrow$ Blending
        MaxScore $\longleftarrow$ Score
return BlendedResultList
```

where $\overline{v}_q$ and $\overline{w}_{qd}$ are, respectively, the vector of source features (see §4.2) and the vector of document features (see §4.3) for query q and document d. This function predicts the *preference* between any pair of documents d_x and d_y coming from either Video or WebVideo. Here, $f(\overline{v}_q, \overline{w}_{qd_x}, \theta) > f(\overline{v}_q, \overline{w}_{qd_y}, \theta)$ means that d_x should be preferred over d_y, i.e., is predicted to be more relevant. Using f, we can easily define a blending algorithm, where the blended result list is obtained by ranking documents according to the predicted preferences and by satisfying the *no re-ordering constraint*; see Algorithm 2. We train f by directly optimizing for the retrieval metric being used to assess the blended result list (e.g., ERR or NDCG); for this purpose, we use YetiRank [8]. We use one copy of each source feature for each source, which yields better performance than using one single feature according to our experiments (as in [3]).

List-wise. In this method, we use a *list-wise* approach to LTR. In other words, we predict the *preference* between any pair of blendings satisfying the *no re-ordering constraint*. This approach is the best fit for our problem and is applicable due to this constraint, which drastically limits the number of possible blendings. Let $f(\overline{v}_q, \overline{u}_b, \theta)$ be a *blending preference function* with parameters θ, where $\overline{v}_q$ and $\overline{u}_b$ are, respectively, the vector of source features (see §4.2) and the vector of blended result list features (see §4.4) for a query q and blending b. This function predicts the *preference* between any pair of blendings b_x and b_y from all possible blendings satisfying the constraint mentioned at beginning of §3. Here, $f(\overline{v}_q, \overline{u}_{b_x}, \theta) > f(\overline{v}_q, \overline{u}_{b_y}, \theta)$ means that b_x should be preferred over b_y, i.e., is predicted to be more relevant. Using f, we can easily define a blending algorithm, where the blended result list is obtained by using the most preferred blending as predicted by f. This algorithm is similar to Algorithm 1, but, in this case, the score of each blending is computed by f directly using blended result list features and without document features. We train f using GBRT [7], where the target score of each blending in the training data is computed using the retrieval metric used.

Algorithm 2: Pair-wise LTR blending algorithm

```
input  : Result list from $S$ for $S \in \{\text{Video}, \text{WebVideo}\}$, $f$, $\theta$ ,$\overline{v}_q$, $\overline{w}_{qd}$, $N$
output: Blended result list
BlendedResultList $\longleftarrow$ {} i $\longleftarrow$ 0 j $\longleftarrow$ 0
for _ $\longleftarrow$ 0 to $N-1$ do
    $d_w \longleftarrow$ ResultList$_{\text{WebVideo}}$[i] // $i^{th}$ result from WebVideo
    $d_v \longleftarrow$ ResultList$_{\text{Video}}$[j]    // $j^{th}$ result from Video
    if $f(\overline{v}_q, \overline{w}_{qd_w}, \theta) > f(\overline{v}_q, \overline{w}_{qd_v}, \theta)$ then
        BlendedResultList[i + j] = $d_w$
        i $\longleftarrow$ i + 1
    else
        BlendedResultList[i + j] = $d_v$
        j $\longleftarrow$ j + 1
return BlendedResultList
```

4 Features

We present the features used in the blending methods proposed in §3.

4.1 Query-source features (source quality-based) These are the features for a given query-source pair used by a Yandex system to decide which vertical results to present and how to integrate them into the Web results. More than one hundred features are currently used by this system, including, in particular, click-through, vertical-intent and hit count features (similar features were used in [1, 2]). For space reasons, we do not describe them here and instead, in §6.2, we discuss only the strongest features. We also use the following features, where the ones from [15] are marked with *: ListMax$_S$, ListMin$_S$, ListMean$_S$*, ListStd$_S$*, ListSkew$_S$*, which are respectively, the maximum, minimum, mean, standard deviation and skew of original documents scores the result list taken from each source S.

4.2 Query-source features (LTR) In our LTR approaches, we use all the features described in 4.1, as well as the following features: PredictedERR@N_S: the predicted value of ERR@N for each source S as computed by q_S (§3.2), where $N \in \{1, 3, 5, 10\}$.

4.3 Query-source-document features These are the features for a given query-source-document triplet (q, S, d), with features from [15] marked with *: (1) Score*: the original document score returned by S. (2) Rank*: the position of d in the results from S. (3) NormScore$^*_{[0,1]}$: the score of d returned by S after the top 10 results were shifted and scaled in the interval $[0, 1]$ using MinMax normalization. (4) IsTopN*: a binary feature to indicate if d is within the top N results, where $N \in \{1, 3, 5, 10\}$. (5) IsVideo: a binary feature to indicate if S is Video.

4.4 Blended result list features These are the features for a given *blended* result list b: (1) List*: the same features as defined in §4.1, but over b. (2) NumFrom{Video, WebVideo}: the number of results from each source in b. (3) CosineSimilarity(S, b): the cosine similarity [12] between each source S and b. We do not use the KL similarity from [12], because it is not well defined in our case. (4) CombinedPredictedERR@N: the combined predicted value of ERR for the top N results ($\overline{\text{ERR}@N_S}$) from each source S from [12], where $N \in \{1, 3, 5, 10\}$, as: $\text{sim}_{cosine}(\text{Video}, b) \cdot \overline{\text{ERR}@N_{\text{Video}}} + \text{sim}_{cosine}(\text{WebVideo}, b) \cdot \overline{\text{ERR}@N_{\text{WebVideo}}}$.

5 Experimental setup

5.1 Research questions In §6.1, we investigate the following observational question: (RQ1) To what extent do the Video and WebVideo complement each other as sources for answering the video intent in terms of user satisfaction? When investigating which results from the two sources should be presented on the SERP in §6.2, we investigate the following research questions: (RQ2) How effective are source quality-based approaches when compared against common federated search methods to solve this problem? (RQ3) How effective are LTR approaches when compared against common federated search methods to solve this problem?

5.2 Data set We randomly sampled 1,000 queries from the live stream of search queries to Yandex that had Video results aggregated into the SERP. For each query, the top 10 results of the Video and WebVideo sources were independently assessed for relevance by one judge using a 5-grade system (*Perfect, Excellent, Good, Fair, Bad*) and the same assessor instructions (video intent) for both types of content.

5.3 Oracle baselines and comparisons We use the following *oracle* blending methods, which utilize relevance judgments of each document and satisfy the *no re-ordering constraint*: (1) *Best uniform blending:* We use the best uniform blending for all queries. (2) *Best blending:* We show the best blending for each query. This method is the upper-bound of any blending method. (3) *Best source:* We show results only from the best source for each query. Here, the *best* (blending, source) refers to the one giving the best results in terms of the retrieval metric being used. We compare all methods mentioned so far, plus those listed in §3, and the Video vertical (i.e., only return results from this vertical) against the WebVideo source (the best source on average). Comparisons are based on average performance across all queries using 10-fold cross-validation.

5.4 Metrics and significance testing We use the well-known ERR and NDCG measures as metrics. Statistical significance is tested using a one-tailed paired t-test (at the $p < 0.05$ level). Regarding the k parameter of the *Source-KMeans* blending method, we tuned it manually using the training data and use $k = 10$ in all our experiments.

6 Results

6.1 Observational We start with our observational research question (RQ1). Let us first give an overview of the results returned by each source. Results from Video include relevant documents (at least *Fair*) from 481 hosts with 71% of the documents coming from the top 5 hosts, while results from WebVideo include relevant documents (at least *Fair*) from 1,850 hosts with only 536 of them from the top host. The intersection between the top 10 results of each source is really small: 68% of queries have no intersection, 17% have one common document, 9% have two common documents, and 6% have three or four common documents (never more). In other words, the two sources seem quite different, which is unsurprising considering the specialized nature of Video in comparison with WebVideo.

Let us now compare the performance of each source (Video and WebVideo) with the oracle blending methods from §5.3. The results are shown in Table 1. Video gives

Table 1: Relative effectiveness of the Video vertical and oracle blending methods when compared against WebVideo. A ▲ (▼) denotes significantly better (worse) performance compared to *(Oracle) Best source*. Bold face indicates best performance per metric.

	ERR@5	ERR@10	NDCG@5	NDCG@10
Video	−5.16%▼	−0.57%▼	−2.68%▼	−1.34%▼
WebVideo	–	–	–	–
(Oracle) Best uniform blending	+4.05%▼	+9.63%▼	+3.81%▼	+7.82%▼
(Oracle) Best blending	**+26.06%**	**+26.65%**	**+33.27%**	**+37.29%**
(Oracle) Best source	+22.25%▼	+22.58%▼	+28.61%▼	+30.77%▼

−5.16% in ERR@5 and −2.68% in NDCG@5 compared to WebVideo, which means that the specialized video vertical is worse, in general, at answering the video intent. When blending results from both sources, the oracle method, as the upper bound, gives +26.06% in ERR@5 and +33.27% in NDCG@5 (RQ1). This large improvement shows that, indeed, the two sources complement each other and that both are required in order to best answer the video intent. The best uniform blending mix (V_1,W_1,V_2,V_3,W_2 for ERR@5) gives only +4.05% in ERR@5 and +3.81% in NDCG@5, so a blending algorithm must be query dependent to get closer to the upper bound. Using the best source for each query gives as much as +22.25% in ERR@5 and +28.61% in NDCG@5.

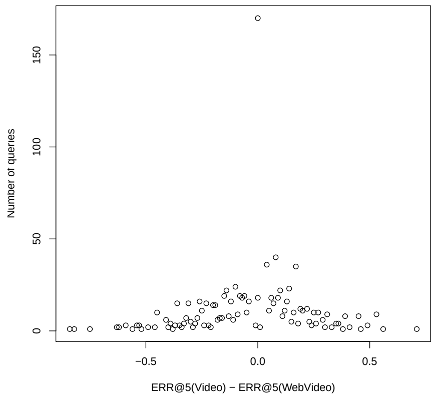

Fig. 2: Number of queries vs. difference in ERR@5 between Video and WebVideo.

For a significant number of queries, the difference between the relevance of the top results from each source is quite small, as shown in Fig. 2 (notice 170 queries at 0). Moreover, as results from the sources are different, presenting results from both sources will lead to an increase in *diversity*. In such cases, e.g., for "*games of thrones watch video*," each source contains results for different sub-intents of the main video intent and a single source would therefore likely lead to user dissatisfaction. Hence, presenting results from both sources while keeping the quality of the blending at the same level is better when each has highly relevant results (RQ1); below, we examine how diverse the blended result lists are for the blending methods from §3.

We also find that, in some cases, one of the two sources does not provide highly relevant results even for queries with a strong dominant video intent as shown in Fig. 2 and Table 3. E.g., for "*watch anime video online*," where Video does not return highly relevant results as opposed to WebVideo due to the query being really ambiguous. Therefore, by itself the intent of the query is not enough to properly decide when to present results from a particular video source, and the relevance of the results w.r.t. sub-intents of the video intent underlying the query must be taken into account.

Table 2: Relative effectiveness of blending methods when compared against WebVideo. A $^\blacktriangle$ ($^\blacktriangledown$) denotes significantly better (worse) performance compared to *Source-KMeans*. Bold face indicates best score per metric.

	ERR@5	ERR@10	NDCG@5	NDCG@10
Round-Robin	+1.90%$^\blacktriangledown$	+7.20%$^\blacktriangledown$	+2.43%$^\blacktriangledown$	+4.44%$^\blacktriangledown$
Raw Score Merging	+0.02%$^\blacktriangledown$	+0.27%$^\blacktriangledown$	+0.21%$^\blacktriangledown$	+1.75%$^\blacktriangledown$
CORI-Size	+3.96%$^\blacktriangledown$	+10.71%$^\blacktriangledown$	+4.69%$^\blacktriangledown$	+9.71%$^\blacktriangledown$
CORI-ML	+8.85%$^\blacktriangledown$	+13.73%$^\blacktriangledown$	+10.76%$^\blacktriangledown$	+15.20%$^\blacktriangledown$
Source-Binary	+12.44%	+13.49%$^\blacktriangledown$	**+15.73**%	+17.01%$^\blacktriangledown$
Source-KMeans	**+13.19**%	**+14.88**%	+15.59%	**+18.07**%
LTR-Pointwise	+11.30%$^\blacktriangledown$	+14.01%	+14.11%$^\blacktriangledown$	+17.14%
LTR-Pairwise	+11.87%$^\blacktriangledown$	+14.82%	+13.80%$^\blacktriangledown$	+17.38%
LTR-Listwise	+11.88%$^\blacktriangledown$	–	+15.20%	–

6.2 Algorithmic Next, we turn to our algorithmic research questions (RQ2, RQ3). The results obtained are shown in Table 2. Of the federated search and fusion-based approaches from §3.1, *CORI-ML* performs best and gives +8.85% in ERR@5 and +10.76% in NDCG@5. Next, our source quality-based approaches from §3.2 perform well as *Source-KMeans* (the best one) gives a +13.19% in ERR@5 and +15.59% in NDCG@5, thus outperforming *CORI-ML* by a large margin (RQ2). The fact that *Source-KMeans* is not significantly better than *Source-Binary* @5 can be explained by the fact that even queries with small predicted differences in quality of the two sources can have really different best blendings, especially close to 0. Our LTR approaches perform well, but do not outperform *Source-KMeans*. Our best LTR approaches are *LTR-Listwise* and *LTR-Pairwise*, which yield +11.88% and +11.87% in ERR@5 and +15.20% and +14.82% in NDCG@5, respectively (RQ3). Results at @10 are consistent with these findings, although *LTR-Listwise* was not used @10 due to the combinatorial explosion. Even though there is no significant difference between the three LTR methods, *LTR-Pointwise* scales much better as the number of sources increases due to the fact that two independent models are used, and its training process is much faster.

Table 3: Content-type diversity of the top 5 results produced by each of the blending methods (using ERR@5).

	The number of results @5 from video hosting sites					
	0	1	2	3	4	5
Video	0%	0%	0%	0%	0%	100%
WebVideo	44%	21%	23%	7%	4%	1%
(Oracle) Best blending	11%	11%	10%	10%	12%	46%
Round-Robin	0%	0%	26%	47%	21%	6%
Raw Score Merging	44%	21%	23%	8%	3%	1%
CORI-Size	3%	19%	24%	26%	20%	8%
CORI-ML	14%	20%	21%	19%	17%	9%
Source-Binary	29%	12%	10%	3%	2%	44%
Source-KMeans	23%	12%	12%	7%	13%	33%
LTR-Pointwise	28%	12%	15%	10%	10%	25%
LTR-Pairwise	26%	13%	14%	10%	8%	29%
LTR-Listwise	19%	14%	11%	11%	15%	30%

Next, we examine content-type diversity of the top 5 results produced by each of the blending methods, i.e., the percentage of result lists with 0 to 5 results from video hosting sites (using ERR@5). The results are shown in Table 3. The content-type diversity of all our source quality-based and LTR approaches is similar. Compared with the oracle method as the upper bound, we observe that our methods have a similar top 5 content-type diversity, but return slightly more result lists with 0 results from video hosting sites and fewer with 5 results. Investigating ways to directly take diversity into account in our blending methods is future work.

Finally, we turn to the contribution of individual features. The top 10 features according to their weighted contribution to q_{Video} from §3.2 (one of the two models used in *Source-KMeans*) using ERR@5 are shown in Table 4; see [9, §10.13] for the description of those weights. We observe that List* features perform really well, which is consistent with previous studies [20]. Other well-performing features are: the number of documents returned by the Video vertical, WebDCG (i.e., DCG of original document scores from WebVideo), IsFilm (i.e., whether the query is about a film), VideoSites (i.e., whether the sites in the top results are known to be relevant for video queries) and VideoTop3MaxRel (i.e., the maximum of the top 3 original document scores from Video). Results for q_{WebVideo} (the second model used in *Source-KMeans* and *Source-Binary*) are consistent with these findings.

Table 4: Top 10 features according to their contribution to q_{Video} from §3.2 (ERR@5).

Rank	Feature	Score	Rank	Feature	Score
1	ListMax	6.70	6	ListSkew	3.56
2	ListMean	6.00	7	WebDCG	3.04
3	ListMin	4.54	8	IsFilm	2.55
4	ListStd	4.41	9	VideoSites	2.51
5	NumDocs(Video)	4.40	10	VideoTop3MaxRel	2.32

7 Conclusion

We have introduced and studied a setting where vertical and Web results are *blended* into a single result list. We have focused on video intent and presented a detailed observational study of a Yandex's two video content sources. The two sources that we consider complement each other and both are required in order to best answer video information needs. An oracle-based upper bound gives +26.06% improvement in ERR@5 and +33.27% in NDCG@5 compared to the case where only WebVideo (the best performing source) is used. For a large number of queries, presenting results from both sources is better for diversity than using a single source.

These observations motivated us to investigate the following algorithmic problem: which results from the two sources should be presented on the SERP? We proposed source quality-based methods and learning to rank methods to address this problem. Our source quality-based approaches perform well (+13.20% in ERR@5 for the best one) and outperform the best baseline approach, a variation of CORI, by a large margin (+8.85% in ERR@5). Our LTR approaches perform well (+11.88% in ERR@5 for the best one), outperforming the best baseline approach but not the source quality-based approaches. Compared with the oracle method as our upper bound, we observe that our methods have a similar top 5 content-type diversity.

Acknowledgments. This research was partially supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement nr 288024 (LiMoSINe project), the Netherlands Organisation for Scientific Research (NWO) under project nrs 640.-

004.802, 727.011.005, 612.001.116, HOR-11-10, the Center for Creation, Content and Technology (CCCT), the QuaMerdes project funded by the CLARIN-nl program, the TROVe project funded by the CLARIAH program, the Dutch national program COMMIT, the ESF Research Network Program ELIAS, the Elite Network Shifts project funded by the Royal Dutch Academy of Sciences (KNAW), the Netherlands eScience Center under project number 027.012.105 and the Yahoo! Faculty Research and Engagement Program.

8 Bibliography

[1] J. Arguello, J. Callan, and F. Diaz. Classification-based resource selection. In *CIKM '09*, pages 1277–1286. ACM, 2009.

[2] J. Arguello, F. Diaz, J. Callan, and J.-F. Crespo. Sources of evidence for vertical selection. In *SIGIR '09*, pages 315–322. ACM, 2009.

[3] J. Arguello, F. Diaz, and J. Callan. Learning to aggregate vertical results into web search results. In *CIKM '11*, pages 201–210. ACM, 2011.

[4] J. P. Callan, Z. Lu, and W. B. Croft. Searching distributed collections with inference networks. In *SIGIR '95*, pages 21–28. ACM, 1995.

[5] S. Dumais, E. Cutrell, and H. Chen. Optimizing search by showing results in context. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 277–284. ACM, 2001.

[6] E. A. Fox and J. A. Shaw. Combination of multiple searches. In *TREC 1993*, pages 243–243. NIST, 1994.

[7] J. H. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232, 2001.

[8] A. Gulin, I. Kuralenok, and D. Pavlov. Winning the transfer learning track of Yahoo!'s learning to rank challenge with YetiRank. *J. Machine Learning Research*, 14:63–76, 2011.

[9] T. Hastie, R. Tibshirani, and J. J. H. Friedman. *The elements of statistical learning*, volume 1. Springer New York, 2001.

[10] K. Kwok, L. Grunfeld, and D. Lewis. TREC-3 ad-hoc, routing retrieval and thresholding experiments using PIRCS. In *TREC 1993*, pages 247–247. NIST, 1995.

[11] M. Lalmas. Aggregated search. In *Advanced Topics in Information Retrieval*, pages 109–123. Springer, 2011.

[12] G. Markovits, A. Shtok, O. Kurland, and D. Carmel. Predicting query performance for fusion-based retrieval. In *CIKM '12*, pages 813–822. ACM, 2012.

[13] A. Ponnuswami, K. Pattabiraman, Q. Wu, R. Gilad-Bachrach, and T. Kanungo. On composition of a federated web search result page. In *WSDM '11*, pages 715–724. ACM, 2011.

[14] Y. Rasolofo, F. Abbaci, and J. Savoy. Approaches to collection selection and results merging for distributed information retrieval. In *CIKM '01*, pages 191–198. ACM, 2001.

[15] D. Sheldon, M. Shokouhi, M. Szummer, and N. Craswell. LambdaMerge: merging the results of query reformulations. In *WSDM '11*, pages 795–804. ACM, 2011.

[16] M. Shokouhi and L. Si. Federated search. *Foundations and Trends in Information Retrieval*, 5(1):1–102, 2011.

[17] M. Shokouhi and J. Zobel. Robust result merging using sample-based score estimates. *ACM Transactions on Information Systems (TOIS)*, 27(3):14, 2009.

[18] C. G. Snoek and M. Worring. Concept-based video retrieval. *Foundations and Trends in Information Retrieval*, 2(4):215–322, 2008.

[19] K.-T. T. Tjin-Kam-Jet and D. Hiemstra. Learning to merge search results for efficient distributed information retrieval. In *DIR '10*, 2010.

[20] V. Vinay, I. J. Cox, N. Milic-Frayling, and K. Wood. On ranking the effectiveness of searches. In *SIGIR '06*, pages 398–404. ACM, 2006.

[21] E. M. Voorhees, N. K. Gupta, and B. Johnson-Laird. Learning collection fusion strategies. In *SIGIR '95*, pages 172–179. ACM, 1995.