

通用纠错系统

拼写纠错（Error Correction，以下简称 EC）是用户比较容易感知的一个功能，借助纠错技术不仅提升了用户体验，还能够挽回流量损失，提升用户粘度。如百度的纠错功能如下图所示



1. EC 简介

1.1 Error Correction (EC)

EC 其实是属于 Query Rewrite（以下简称 QR）模块中的一个功能，QR 模块包括拼写纠错，同义改写，关联 query 等多个功能。QR 模块对于提升用户体验有着巨大的帮助，对于搜索质量不佳的 query 进行改写后能返回更好的搜索结果。QR 模块内容较多，以下着重介绍 EC 功能。

当用户看到搜索结果较差较少时，如果能意识到自己的 query 错误，对 query 进行修正再次检索，也许能找到想要的结果。但有时用户也不知道自己的 query 错在哪里，这时就需要纠错服务为用户提供纠错结果。

1.2 EC 形式

一类是单词拼写错误，早期的英文串纠错就是通过字典进行单词正确性的纠错，如将“happy”拼成“hbppy”；第二类是上下文搭配不当引起的错误。在中文由于最小语义单元是字，所以不存在第一类的错字情况，只有字与字搭配的词条是否合理，或者一连串词条搭配能否组成一个通顺的句子，所以中文纠错主要解决的是上下文搭配问题。比如上班清代

1.3 EC 类型

错误类型	纠错示例
数字	2408->2048
英文	Fiappy->flappy Whatasapp->whatsapp Justwayyouare ->just the way you are
拼音	Talang->踏浪 wangfei ->王菲
简拼	Zgr->中国人/张国荣 Xiaopg->小苹果
缺字	手机助->手机助手 倍爽->倍儿爽
多字	你是我的眼毛->你是我的眼 送情郎当红军->送郎当红军
错字	笨鸟学飞->笨鸟快飞

汉字拼音	朗朗->郎朗 草冒歌->草帽歌
模糊音	四面埋伏->十面埋伏
拼音+英文	2014zuixindj->2014 最新 dj
汉字+英文	江南 st-> 江南 style
汉字+拼音	情非得已 tongs->情非得已童声
换序	老师对话->对话老师 宗雨林->雨宗林
形近字	许嵩精选->许嵩精选 中固话->中国话
关联	时间都去哪了 王铮-> 时间都去哪儿了 王铮亮 阿杜擦一点 ->阿杜差一点
Part 纠错	非主流 dj 背尽音乐 ->非主流 dj 背景音乐

2.通用 EC 设计

2.1 系统流程

纠错流程一般来说分为

1 、Query 纠错判断。对于常见错误，例如常见的拼写错误，使用事先挖掘好的错误 query 字典，当 query 在此字典中时纠错。如果用户输入的 query 查询无结果或结果较少于一定阈值时，尝试纠错，可以根据不同领域的策略和容忍度，配置最少结果数阈值。

2、不同策略独立纠错。一般来说使用拼音纠错和编辑距离纠错，并辅助模糊音形近字二次纠错等其他纠错策略比较常见。

同音策略是用户输入的错误 query 和候选纠错 query 有相同的拼音。

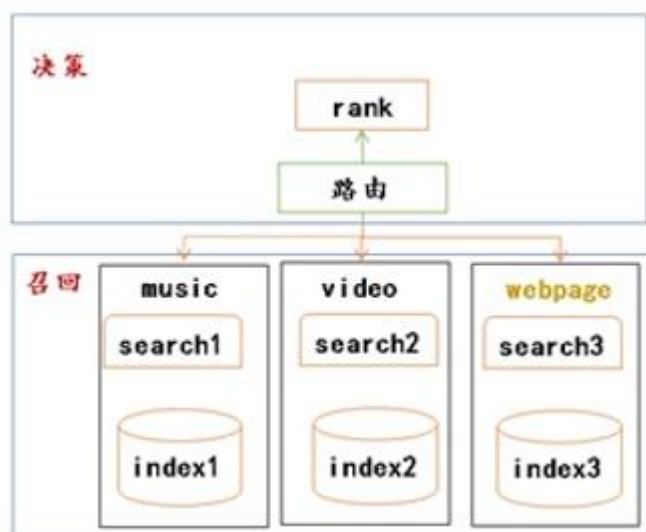
编辑距离策略就是错误 query 和候选 query 之间编辑距离小于一定阈值，并配合其他条件进行过滤。

3、候选词结果选择。因为每个策略比较独立，不同策略会给出不同的候选词，因此对于候选词的选取，每个策略有所不同。不同策略之间，不同策略内部需要使用不同的评估方式，来选择最优结果。

2.2 系统通用

从两个角度考虑通用性：首先各个资源集群可以复用，上层可以根据业务需求，路由调用多个集群的数据索引；其次是各个业务做到可定制，这样就需要将不同业务的策略做到可配置。

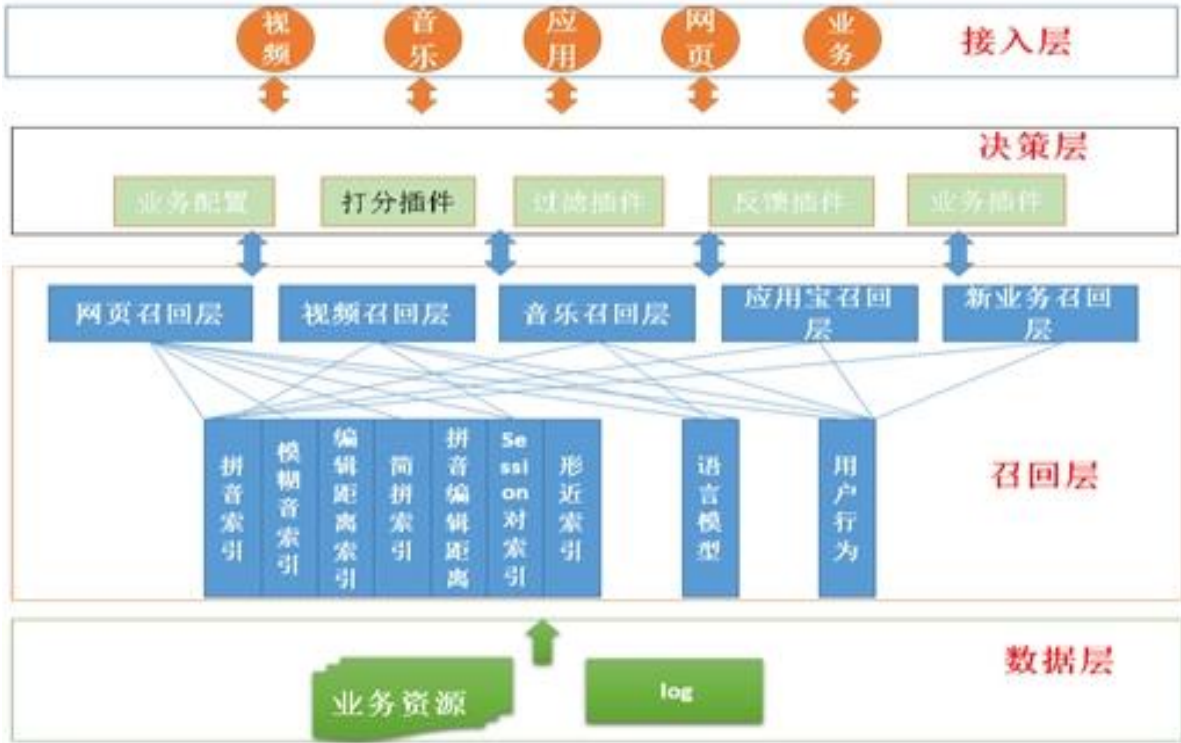
通用 EC 将其改为召回和排序两个线上服务，就可以满足上面的两个通用性要求。例如，音乐业务，需要调用 music 和 webpage 两套集群；视频业务需要调用 video 和 webpage；泛娱乐的业务，需要调用 music, video, webpage 三套集群。新的系统只需要各自搭建一套就可以满足需求；



2.3 分层设计

逻辑上分为四个层次：

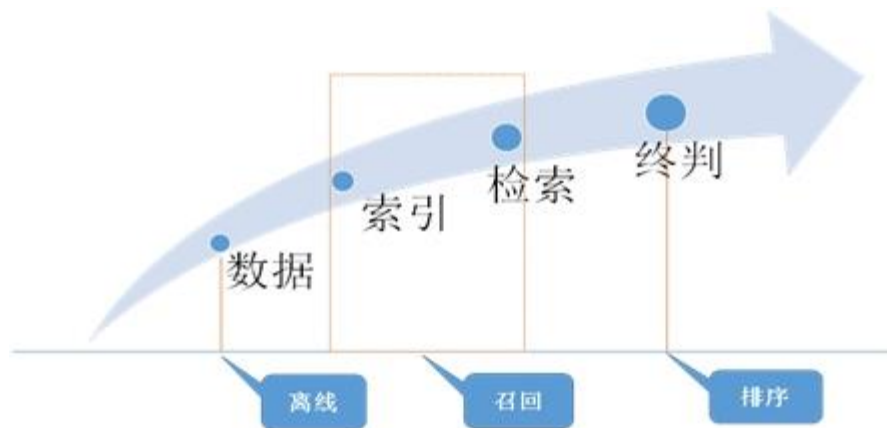
接入层	负责业务接入的接口；
决策层	对各个业务进行定制化的排序服务；各种计算插件；
召回层	数据索引构建；在线检索召回；
数据层	数据筛选和去噪；离线纠错对挖掘；运营指标统计和监控；在线反馈。



图表 3 通用 EC 逻辑图

2.4 系统模块

纠错系统大致分为几个部分，包括数据筛选和去噪，索引构建，在线检索，对结果进行终判得到最终结果。可以归纳为三部分：离线数据处理，数据召回和排序终判。



1. 数据模块对搜索 log 定期进行抽取和统计,对 query 进行归一化后给出 query 频次词典。对数据库信息整理给出自定义词典。通过爬虫系统爬取优质词条词典
2. 离线数据处理使用数据模块准备好的各种词典生就纠错词典, 包括拼音纠错词典, 编辑距离纠错词典等。根据配置, 对频次词典中对超出一定长度 query 上述操作不处理
3. 在线检索负责 query 实时纠错: 如果第一次纠错 query 查询结果较差, 使用扩大召回的方式, 比如二次纠错、片段纠错等扩大召回重新纠错, 进行二次查询并返回质量较高的查询结果
4. 纠错效果的好坏从微观上讲, 可以查看搜索日志中无结果或结果少的纠错 query 以及点击模型中点击较少的纠错 query 等方式发现 badcase, 再针对这些 badcase 出现的原因进行分类总结。在宏观上可以关注搜索效果评估系统中的 MR 和 MRR 分数, 使用 AB Test, 查看使用纠错模块后带来的效果提升

3. 算法层面

3.1 噪音信道模型

拼写纠错的过程, 是用户在书写的过程中通过了“噪声信道”, 导致了错误的拼写形式, 这个“噪声信道”不是传输介质, 而是输入法, 键盘, 用户

大脑(拼音错误理解或者表达记忆不准确)，如何去还原出正确的 query。这个模型作为最基本的纠错原理。



$$I = \underset{I}{\operatorname{argmax}} P(I|O) = \underset{I}{\operatorname{argmax}} \frac{P(O|I)P(I)}{P(O)} = \underset{I}{\operatorname{argmax}} P(O|I)P(I)$$

3.2 编辑距离

编辑距离模型是指通过基本的“插入”、“删除”、“替换”操作，来计算两个串之间的距离，从而衡量二者的相似度。举例来讲，apple 和 apply 的编辑距离是 1，access 和 actress 的编辑距离是 2，arrow 和 brown 的编辑距离是 3。在 DNA 分析，图像识别，抄袭侦测等方面应用广泛。

表格 2 编辑距离示例

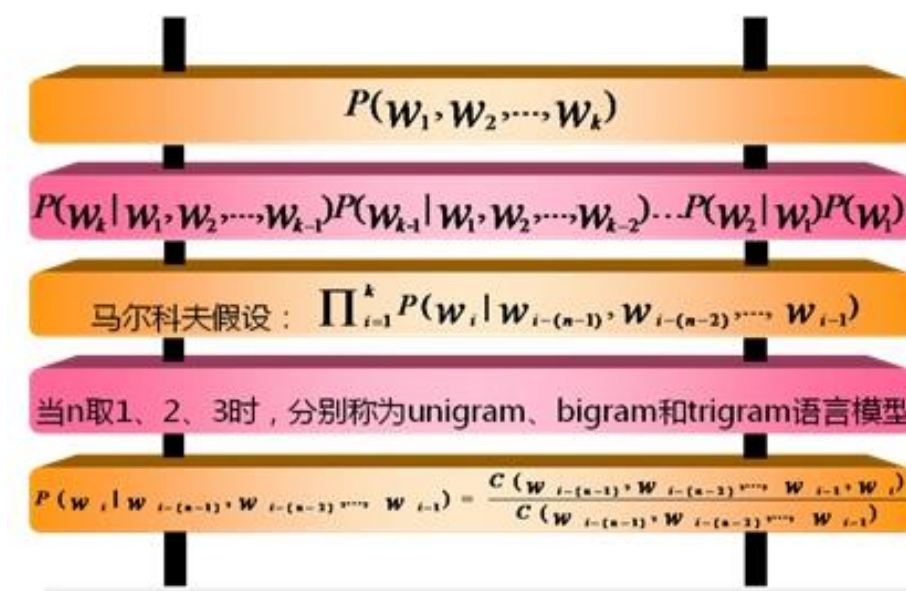
	0	民	主	共	和	国
0	0	1	2	3	4	5
人	1	1	2	3	4	5
民	2	1	2	3	4	5
共	3	2	2	2	3	4
和	4	3	3	3	2	3

主	5	4	3	4	3	4
意	6	5	4	4	4	4

3.3 语言模型

统计语言模型是给词语串指派概率的方法，不论是计算整句话的概率，还是在一个序列中估计下一个将要出现的词语串概率，都会是用到语言模型。

信道噪音模型中 $P(I)$ 和 $P(O)$ ，指候选串和原始串中词语搭配的概率。片段纠错过程中，也是基于语言模型计算各个拼接候选结果是否合理。



图表 4 语言模型

3.4 搜索 Session 的挖掘

用户在搜索过程往往是一个序列，在错误的情形下，很多用户可能在序列中将错误串主动进行改正。利用用户的主动修改行为，可以产生更为高效和精准的候选纠错对。搜索 session 指的是用户在某一个时间段内的搜索行为，如果把搜索日志按照时间排序，对于某一个用户的搜索日志来说，可以看到用户的

搜索行为是分段的，每段之间往往有较为明显的间隔，每一段我们称为一个搜索 session。

点击模型中的一些统计数据可以判断一个搜索 query 质量的高低，质量高的 query 往往会给出较好的结果，用户点击的欲望更高。举例来说，“度假”（正确）“渡假”（错误）这两个 query，假设用户输入较多的是错误的“渡假”，系统给出结果会比较差。在这种情况下，虽然“渡假”的搜索次数更多，但是点击模型给出 query 分数会比较低，而候选词“度假”的 query 得分就会高一些，同时可以辅助其他纠错方式完成纠错。

流程如下：



图表 5 纠错对挖掘流程

通用 EC 对纠错对的挖掘做了一些泛化，将模式识别的技术应用于挖掘中，来提高长尾的召回。例如：用户 session 中出现“山西明格->山西民歌”，挖掘出泛化标签“地域明格->地域民歌”；线上可以应用到“山东明格->山东民歌”。

4.策略层面

4.1 召回策略

搜索系统许多功能的召回率和准确率是矛盾的，但是在 query 纠错问题中，准确率往往要求更高。拼音 query 到汉字 query 的纠错，往往会存在较大的转义风险，不同的类型的拼音转换方式（全拼，模糊全拼，简拼，混拼）有着不同程度的转义风险，召回越大则准确率降低，因此使用全拼较为稳妥。

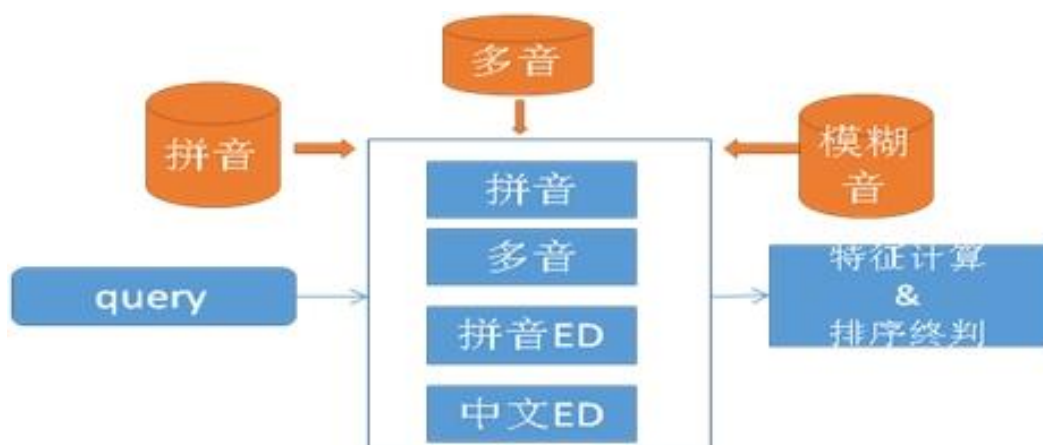
由于网页和垂搜的数据差异，在召回方面存在比较大的差异。网页主要依赖 log 和离线的纠错对；垂搜 log 量少噪音大，主要使用资源数据，通过编辑距离方式将用户的输入串找到最相近的资源数据。具体差异如下：

网页	中文：拼音，模糊音，纠错对+组合 英文：标签，编辑距离，纠错对+组合
垂搜	中文：编辑距离，拼音，纠错对+组合 英文：编辑距离，标签，纠错对+组合

4.2 中文/拼音

中文是音、形、义于一体的。音是发音，现在大部分人使用拼音输入法；形是指在形体上看似相似；义是说用户真正想要查找的东西，有时候同表达存在一定差异。

由于拼音输入法的普及，以及不同地域的发音特点，在一般的注音基础上，还需要做些变换，才能将最好的结果进行召回。流程如下：



图表 6 中文纠错流程

在垂搜业务中，用户的意图往往较为明确，大部分需求是找具体资源名，例如音乐名或者歌手；再加上数据量少，所以主要采用编辑距离的方式进行召回。在计算相关性上，用到的特征包括拼音相似度，声母、韵母相似度，字形的相似度，键盘相似度等。

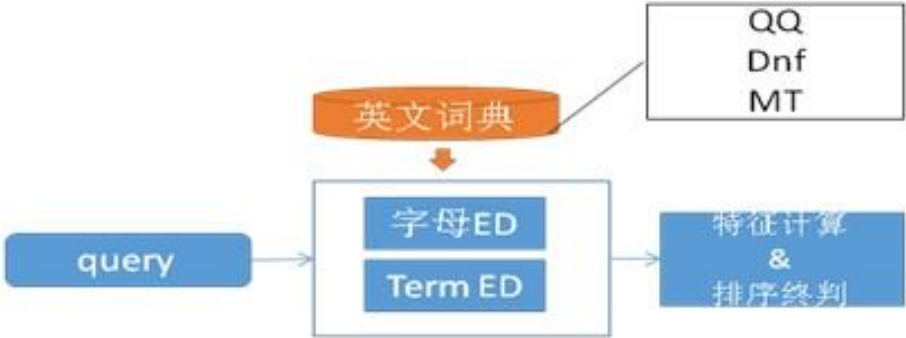
Q	W	E	R	T	Y	U	I	O	P
A	S	D	F	G (i,j-1)	H (i,j))	J (i,j+1)	K	L	
	Z	X	C	V	B	N	M		

图表 7 键盘字符分布

4.3 英文/数字

相比网页搜索，垂搜英文纠错存在两方面特点，一是英文资源占到一定比例，例如一些英文歌曲、中文歌手英文名、app 名字等，英文搜索串比例较高；二是不局限在一般的英文单词纠错，因为有些歌手、歌曲的名称是造出来的，本身就不是一个正确的单词。

英文串的处理流程是：首先将字母编辑距离半径为 r (≤ 2) 的各种组合，在检索库中进行查找，并将候选结果进行匹配计算；如果没有期望的结果，再进行 term 级别的查找。流程如下：



图表 8 英文纠错流程

4.4 关联纠错

1. 新增的纠错形式

以往的纠错形式包括两类，一是英文单词错误，针对的是单个词；二是上下文搭配，针对词组搭配。我们提出的关联纠错，针对的是多个语义块，而每个语义块是正确的，多个语义块在一起是错误的。

2. 关联关系，是指资源对之间通过某种连接存在关系。例如，歌手 A 演唱了歌曲 B，则 A 和 B 之间有关联；演员 C 在某个影视剧 D 中的演员，则 C 和 D 存在关联。

3. 利用关联关系可以有两方面收益。一是提高召回，传统正确的搜索串（甚至人的直观印象也认为是正确的），通过资源关联后，也可以进行纠错；二是提高精度，在终判模块将一些落败的候选结果可以进行保护。

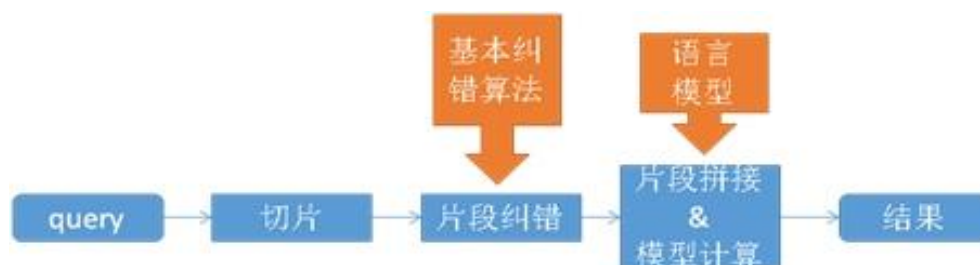
具体流程是，首先将 query 进行切片，切分的原则是尽量不破坏资源实体；其次是选出最有可能的若干片段，对每个片段找出候选结果；最后将候选结果进行组合，根据关联关系找出最好的结果。例如：



图表 9 关联纠错示例

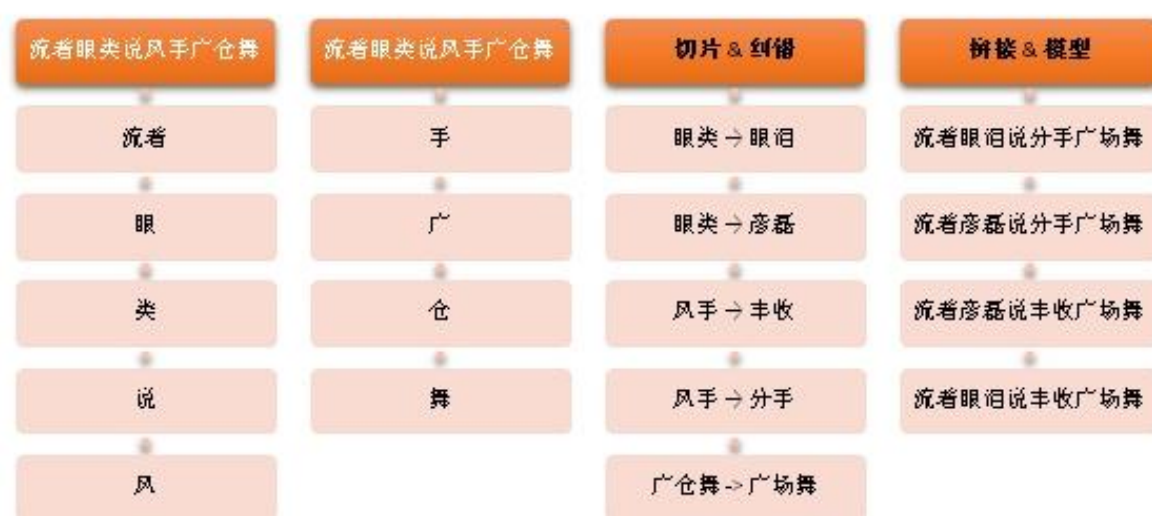
4.5 片段纠错

对于长尾的检索串，log 出现的次数很少甚至没出现过，所以整串纠错难度很大。就需要将 query 串进行拆分，对每个片段根据上面提到的基本纠错算法进行纠错，之后再进行拼接判定。流程如下：



图表 10 片段纠错流程

需要注意几个规则：切分过程中尽量将实体保存在一个片段中，使得拆分片段尽量保留原意；拼接后，使用语言模型进行判定，判定构成一个通顺句子的可能性。示例如下：



4.6 线上策略

线上使用并不是直接将纠错串进行替换，主要有两点原因：一是纠错使用的是先验特征，看不到原串和纠错串的结果信息；二是兼顾不到所有的检索表达式。

例如，在“真实主驾驶”纠错为“真实驾驶”，而正确的结果资源应该是“主驾驶”。纠错表示成用 ABCD→AbCD；而检索表达式会很多样：ABXXCD、丢 A 使用 BCD 检索等。

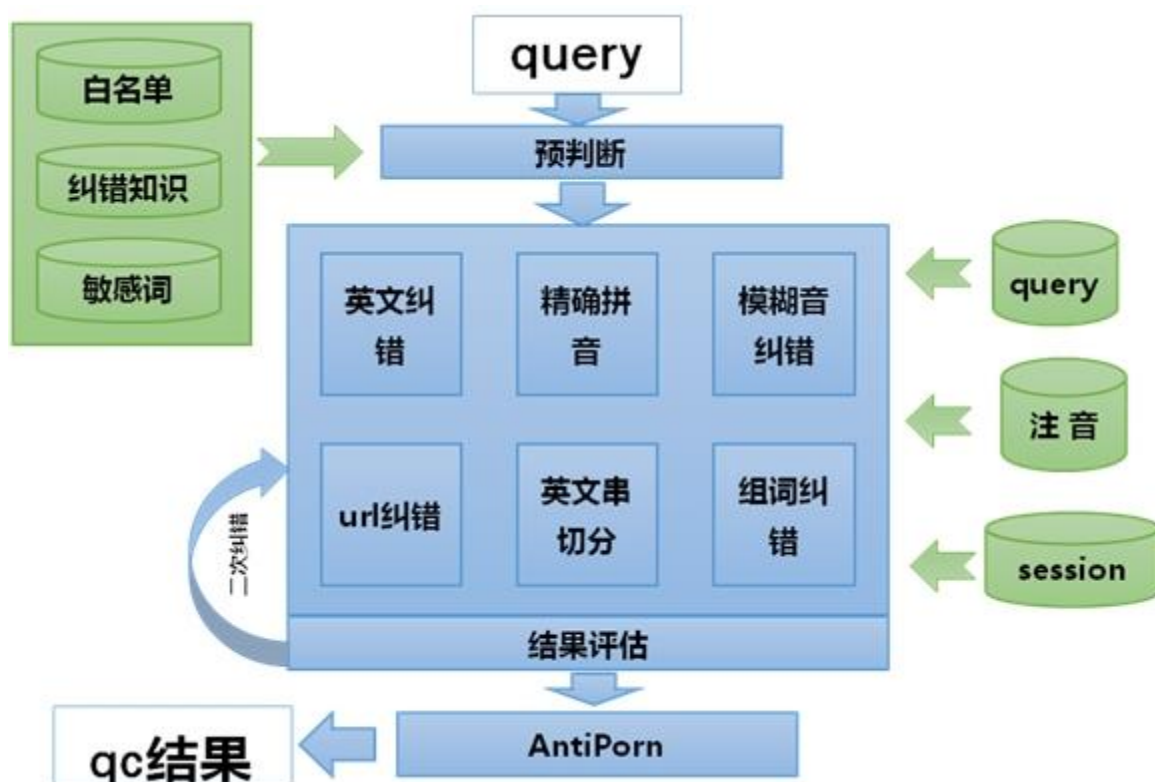
所以采取了一些方案：

1. 策略：不同数据规模/需求强度，采用不同的策略；
2. 展示类型：库内库外/热度差异等，进行混插；置信度低的，进行只提示；
3. 正负反馈：用户点击分析形成正负反馈；
4. 后验融合：原串、纠错串结果比较。

5. 网页&垂搜

5.1 网页 EC

EC 框架图：



图表 1 网页 EC

5.2 垂搜 EC

垂搜 EC 一般在网页纠错服务的基础上，进行浅层开发，不同的业务进行了定制化的服务，包括在线策略、离线挖掘、干预等模块等，开发人员也是各个业务独立的。具体流程如下：



图表 2 垂搜 EC

6.通用纠错难点

6.1 存在问题

6.1.1 效果不理想

一般垂搜是在网页 EC 的基础上做二次加工，但是大部分结果还是基于网页 EC。在调研的过程中，发现网页 EC 很难满足各个业务的需求，问题如下：

错纠	shixiong -> 师兄 （尸兄）
	laobing -> 烙饼 （老兵）
漏纠	罗琦 -> 罗琦
	lqyj1 -> 恋曲一九九零
	老子今天不上班 -> 老子明天不上班

	pround foryou -> pround of you 变形记 湖南卫视 -> 变形计 湖南卫视
过纠	cup -> cpu 百分摔跤 -> 百分摔角

6.1.2 开发冗余

EC 虽然覆盖率小，但每个业务都是必不可少的。各个业务都要单独维护，开发人员也只是解决一些棘手的 case，不会长期、深入的跟进；运营以及维护造成了浪费。

6.1.3 机器浪费


各个业务都需要搭建独立的 EC 服务，机器耗费比较大。

6.2 差异

	垂搜	网页
意图	意图明确，用户需求大部分是具体的资源	意图发散，需要满足所有需求
模型	Log 量少，百万级别 可用于离线挖掘语料稀疏	Log 量巨大，近上亿的数据量

表格 1 意图差异 case

[应用] [原串] [音乐]	<div>消灭星星 ← 消星星 → 小星星</div>
-------------------	-----------------------------

[应用] [原串] [视频]	
[应用] [原串] [视频]	

6.3 挑战

- 1. 业务定制化，共性和差异区分，业务之间不受干扰；
- 2. 高召回，可以使用的 log 量和网页不是一个数量级；
- 3. 高精度，垂搜的用户意图更为明确；
- 4. 产品展示丰富，要求给出的纠错类型更精细。