Word Storms: Multiples of Word Clouds for Visual Comparison of Documents

Quim Castellà
School of Informatics
University of Edinburgh
quim.castella@gmail.com

Charles Sutton
School of Informatics
University of Edinburgh
csutton@inf.ed.ac.uk

ABSTRACT

Word clouds are popular for visualizing documents, but are not as useful for comparing documents, because identical words are not presented consistently across different clouds. We introduce the concept of *word storms*, a visualization tool for analyzing corpora of documents. A word storm is a group of word clouds, in which each cloud represents a single document, juxtaposed to allow the viewer to compare and contrast the documents. We present a novel algorithm that creates a coordinated word storm, in which words that appear in multiple documents are placed in the same location, using the same color and orientation, across clouds. This ensures that similar documents are represented by similar-looking word clouds, making them easier to compare and contrast visually. We evaluate the algorithm using an automatic evaluation based on document classification, and a user study. The results confirm that a coordinated word storm allows for better visual comparison of documents.

Categories and Subject Descriptors

H.5 [**Information Search and Retrieval**]: Information Interfaces and Presentation

1. INTRODUCTION

Because of the vast number of text documents on the Web, there is a demand for ways to allow people to scan large numbers of documents quickly. A natural approach is visualization, under the hope that visually scanning a picture may be easier for people than reading text. One of the most popular visualization methods for text documents are *word clouds.* A word cloud is a graphical presentation of a document, usually generated by plotting the document's most common words in two dimensional space, with the word's frequency indicated by its font size. Word clouds can be easy for naive users to interpret and can be aesthetically surprising and pleasing. One of the most popular cloud generators, Wordle, has generated over 1.4 million clouds that have been publicly posted [8].

Despite their popularity for visualizing single documents, word clouds are not as useful for navigating groups of documents, such as blogs or Web sites. The key problem is that word clouds are difficult to compare visually. For example, say that we want to compare two documents, so we build a word cloud separately for each document. Even if the two documents are topically similar, the resulting clouds can be very different visually, because the shared words between the documents are usually scrambled, appearing in different locations in each of the two clouds. The effect, as we confirm in a small user study (Section 5.3), is that it is difficult to see by eye which words are shared between the documents.

In this paper, we introduce the concept of *word storms* to afford visual comparison of groups of documents. Just as a storm is a group of clouds, a word storm is a group of word clouds. Each cloud in the storm represents a subset of the corpus. For example, a storm might contain one cloud per document, one cloud to represent all the documents written in each year, or one cloud to represent each track of an academic conference. Effective storms make it easy to compare and contrast documents visually. We propose several principles behind effective storms, the most important of which is that *similar documents should be represented by visually similar clouds.* To achieve this, algorithms for generating storms should coordinate the layout of their constituent clouds.

We present a novel algorithm for generating *coordinated* word storms that follow this principle. The goal is to generate a set of visually appealing clouds, under the constraint that if the same word appears in more than one cloud in the storm, it appears in a similar location. Interestingly, this also allows a user to see when a word is *not* in a cloud: simply find the desired word in one cloud and check the corresponding locations in all the other clouds. At a technical level, our algorithm combines the greedy randomized layout strategy of Wordle, which generates aesthetically pleasing layouts, with an optimization-based approach to maintain coordination between the clouds. The objective function in the optimization measures the amount of coordination in the storm, inspired by multidimensional scaling.

We evaluate this algorithm on several text corpora, including academic papers and research grant proposals. First, we present a novel automatic evaluation method for word storms based on how well the clouds, represented as vectors of pixels, serve as features for document classification. The automatic evaluation allows us to rapidly compare different layout algorithms, and may be of independent interest as a framework for comparing visualizations. Second, we present a user study in which users are asked to examine and com-

Copyright is held by the International World Wide Web Conference Committee (IW3C2). IW3C2 reserves the right to provide a hyperlink to the author's site if the Material is used in electronic media.
WWW'14, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2744-2/14/04.
http://dx.doi.org/10.1145/2566486.2567977.

pare the clouds in a storm. Both experiments demonstrate that a coordinated word storm is dramatically better than independent word clouds at allowing users to visually compare and contrast documents.

2. DESIGN PRINCIPLES

A word storm is a group of word clouds constructed to visualize a corpus of documents. In the simplest type of storm, each cloud represents a single document by creating a summary of its content; hence, by looking at the clouds a user can form a quick impression of the corpus's content and analyse the relations among the different documents.

Our work builds on word clouds because they are popular, easy for users to understand, and are often visually appealing. By building a storm based on word clouds, we create an accessible tool that can be readily understood without requiring a background in statistics or text processing. The aim of a word storm is to extend the capabilities of a word cloud: instead of visualizing just one document, it is used to visualize an entire corpus.

There are two design motivations behind the concept of word storms. The first is to *visualize high-dimensional data in a high-dimensional space.* Many classical visualization techniques are based on dimensionality reduction, i.e., mapping high-dimensional data into a low dimensional space. Word storms take an alternative strategy, of mapping high dimensional data into a different high dimensional space, but one which is tailored for human visual processing. The second design motivation is the *principle of small multiples* [16, 17], in which similar visualizations are presented together in a table so that the eye is drawn to the similarities and differences between them. A word storm is a small multiple of word clouds. This motivation strongly influences the design of effective clouds, as described in Section 2.3.

2.1 Types of Storms

Different types of storms can be constructed for different data analysis tasks. In general, the individual clouds in a storm can represent a group of documents rather than a single document. For example, a cloud could represent all the documents written in a particular month, or that appear on a particular section of a web site. It would be typical to do this by simply merging all of the documents in each group, and then generating the storm with one cloud per merged document. This makes the storm a flexible tool that can be used for different types of analysis, and it is possible to create different storms from the same corpus and obtain different insights. Here are some example scenarios:

1. **Comparing Individual Documents**. If the goal is to compare and contrast individual documents in a corpus, then we can build a storm in which each word cloud represents a single document.

2. **Temporal Evolution of Documents**. If we have a set of documents that have been written over a long period, such as news articles, blog posts, or scientific documents, we may want to understand trends in the corpus over time. This can be achieved using a word storm in which each cloud represents a time period, e.g., one week or one month. By looking at the clouds sequentially, the user can see the appearance and disappearance of words and how their importance changes over time.

3. **Hierarchies of Documents**. If the corpus is arranged in a hierarchy of categories, we can create a set of storms, one for each category, each of which contains one cloud for each subcategory. For instance, this structure can be useful in a corpus of scientific papers. At the top level, we would first have a storm that contains one cloud for each scientific field (e.g., chemistry, physics, engineering), then for each field, we also have a separate storm that includes one cloud for each subfield (such as organic chemistry, inorganic chemistry) and so on until arriving at the articles. An example is shown in Figures 2 and 3. For large document collections, it is infeasible for a user to visually scan a large number of clouds. In this setting, a hierarchical approach seems particularly appropriate.

Hereafter we use the term "document" to refer to the text represented by a single cloud, with the understanding that the "document" may have been created by concatenating a set of smaller documents.

2.2 Levels of Analysis of Storms

A word storm allows the user to analyse the corpus at a variety of different levels:

1. **Overall Impression of Corpus**. By scanning the largest terms across all the clouds, the user can form a quick impression of the topics in the corpus.

2. **Comparison of Documents**. The user can visually compare clouds in the storm in order to compare and contrast documents. For example, the user can look for words that are much more common in one document than in another. Also the user can compare whether two clouds have similar shapes, to gauge the overall similarity of the corresponding documents.

3. **Analysis of Single Documents**. Finally, the clouds in the storm have meaning in themselves. Just as with a single word cloud, the user can analyze an individual cloud to get an impression of a single document.

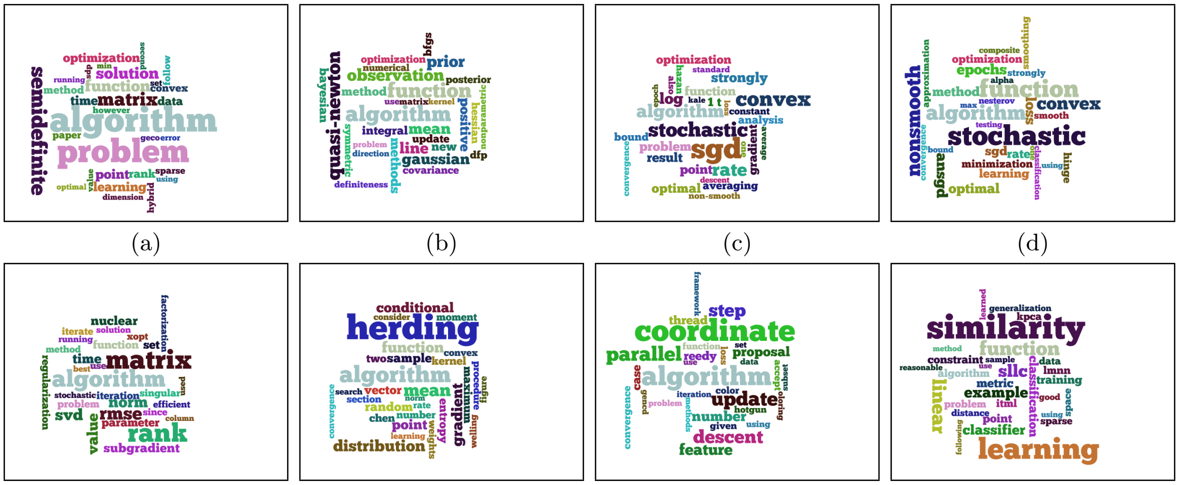
2.3 Principles of Effective Word Storms

Because they support additional types of analysis, principles for effective word storms are different than those for individual clouds. This section describes some desirable properties of effective word storms.

First of all, *each cloud should be a good representation of its document.* That is, each cloud ought to emphasize the most important words so that the information that it transmits is faithful to its content. Each cloud in a storm should be an effective visualization in its own right.

Furthermore, the clouds should integrate harmoniously into a complete storm. In particular, *clouds should be designed so that they are effective as small multiples* [16, 17], that is, they should be easy to compare and contrast. This has several implications. First, clouds should be similar so that they look like multiples of the same thing, making the storm a cohesive unit. Because the same structure is maintained across the different clouds, they are easier to compare, so that the viewer's attention is focused on the differences among them. A related implication is that the clouds ought to be small enough that viewers can analyze multiple clouds at the same time without undue effort.

The way the clouds are arranged and organised on the canvas can also play an important role, because clouds can be more easily compared to their neighbors than to more distant clouds. This suggests the principle that *clouds in a storm should be arranged to facilitate the most important comparisons.* In the current paper, we simply arrange the


optimization second min solution sdp follow running set convex method function semidefinite time matrix data however algorithm paper geco error problem point rank sparse using value optimal learning dimension hybrid
bfgs optimization prior numerical observation posterior bayesian quasi-newton method function use matrix kernel algorithm positive hessian nonparametric symmetric integral mean problem update direction methods line new gaussian dfp covariance definiteness
optimization standard hazan also strongly epoch function log kale 1 t convex loss algorithm constant analysis stochastic bound gradient average problem sgd one result convergence point rate descent optimal averaging non-smooth
smoothing composite approximation optimization epochs strongly alpha nonsmooth method function max nesterov algorithm loss convex smooth testing stochastic convergence bound asgd sgd rate one minimization classification using hinge learning optimal
factorization nuclear iterate solution xopt running function set method time matrix best use regularization algorithm used stochastic iteration singular problem norm efficient svd rmse since column value parameter rank subgradient
conditional consider moment herding function convex two sample kernel algorithm procedure maximum figure search vector mean convergence section norm gradient random rate chen number entropy point welling learning weights distribution
framework step thread coordinate function set parallel reedy loss proposal use data case algorithm accept subset iteration color pscd problem update oloring methods hotgun number given using convergence descent feature
learned generalization kpca similarity method function constraint sample data reasonable use algorithm sllc lmnn classification training metric linear example good problem itml space distance using point sparse following classifier learning

(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 1: Visualization of eight papers from the ICML 2012 conference in a coordinated word storm. These papers appeared in a track about optimization algorithms. A larger version of this storm, which included all accepted ICML papers, was deployed on the official conference Web site during the meeting.

clouds in a grid, but future work could consider methods of organizing the clouds on the canvas.

A final, and perhaps the most important, principle is the *coordination of similarity principle*. In an effective storm, visual comparisons between clouds should reflect the underlying relationships between documents, so that *similar documents should have similar clouds, and dissimilar documents should have visually distinct clouds*. This principle has particularly strong implications. For instance, words should appear in a similar font and similar colors when they appear in multiple clouds. More ambitiously, words should also have approximately the same *position* when they appear in multiple clouds. The coordination of similarity principle can significantly enhance the usefulness of the storm. For example, to compare the most common words in two documents, one can visually check if a word in one cloud also appears in another cloud. Displaying shared words in the same color and position across clouds makes this task much easier, especially when checking for words that appear in one cloud *but not* in another. This principle furthermore tends to encourage the overall shape of the clouds of similar documents to appear visually similar, allowing the viewer to assess document similarity by quickly scanning the clouds.

These principles present new algorithmic challenges. Existing algorithms for single clouds do not consider relationships between multiple clouds in a storm. In the next sections we propose new algorithms for building effective storms.

3. CREATING A SINGLE CLOUD

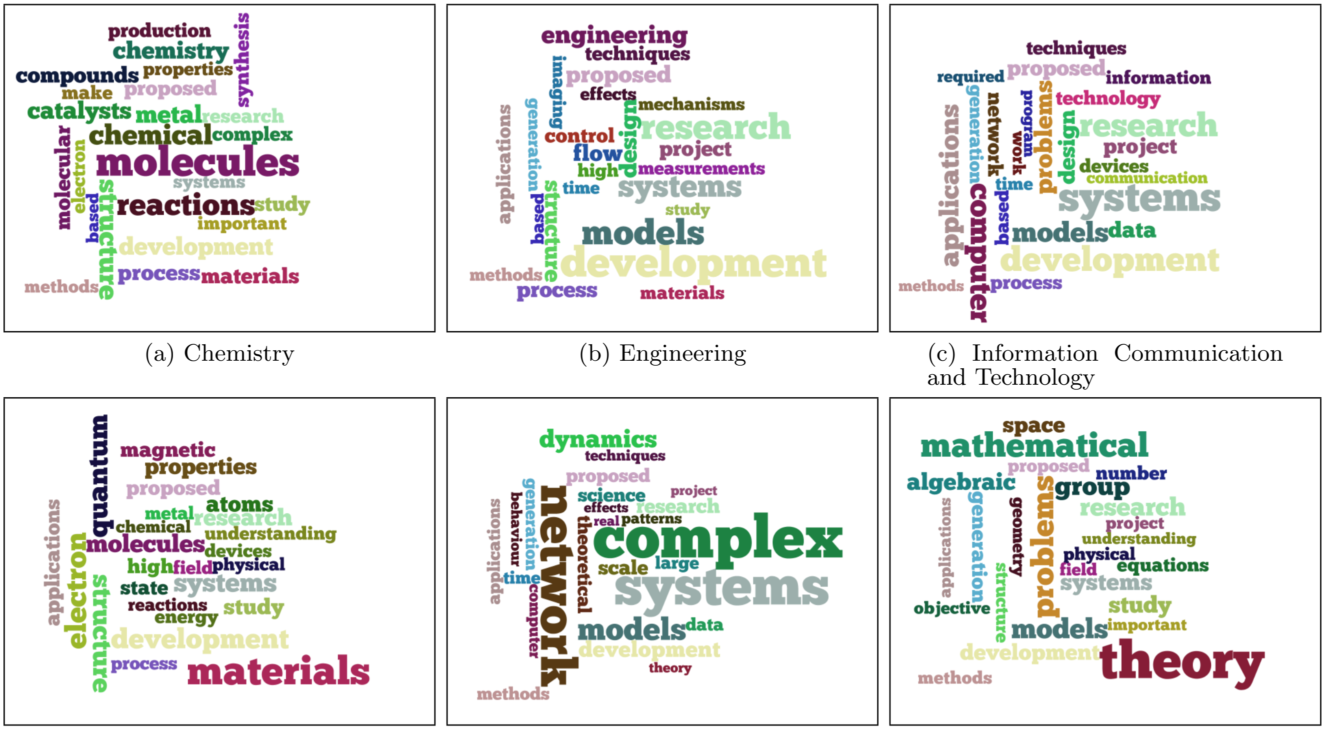
In this section, we describe the layout algorithm for single clouds. The method is based on that of Wordle [8], because it tends to produce aesthetically pleasing clouds. Formally, we define a word cloud as a set of words $W = \{w_1, \ldots, w_M\}$, where each word $w \in W$ is assigned a position $p_w = (x_w, y_w)$ and visual attributes that include its font size s_w, color c_w and orientation o_w (horizontal or vertical).

To select the words in a cloud, we need a measure of the importance of a word to the document, which we call its *weight*. Typically term frequency (*tf*) is used for this, but alternatives could include *idf* with respect to a document collection, or mutual information with respect to document metadata. We select the words in the cloud by choosing the top M words from the document by weight after removing stop words. The font size is set proportionally to the term's weight, and the color and orientation are selected randomly. Choosing the word positions is more complex, because words must not overlap on the canvas. We use the layout algorithm from Wordle [8], which we will call the spiral algorithm.

The spiral algorithm (Algorithm 1) is greedy and incremental; it sets the location of each word in order of size. At the beginning of the i-th step, the algorithm has generated a partial word cloud containing the $i-1$ words of largest weight. To add a word w to the cloud, the algorithm places it at an initial desired position p_w (e.g., chosen randomly). If at that position, w does not intersect any previous words and is entirely within the frame, we go on to the next word. Otherwise, w is moved outwards along a spiral path until it reaches a valid position, that is, a position inside the frame with no overlaps. This is repeated for all words in the cloud.

As the algorithm assumes that the size of the frame is given, we estimate the necessary width and the height to fit M words. Similarly, if the initial desired positions are not given, we sample them from a Gaussian distribution with mean at the frame's center. This distribution is truncated so that the desired position is always sampled within the frame. Notice in line 4 of the algorithm that a maximum number of iterations is used to prevent words from looping forever, which can happen if the word cannot be fit into the frame. If the maximum number of iterations is reached for any word, we assume that the current frame is too small and restart the algorithm with a larger frame.

It is better to check if two words intersect at the glyph level, rather than using a bounding box around the word, to ensure a compact result. However, checking the intersection of two glyphs can be expensive, so instead we use a tree of rectangular bounding boxes that closely follows the shape of the glyph, as in [8]. We use the implementation from the open source library WordCram (`http://wordcram.org`).



(a) Chemistry

(b) Engineering

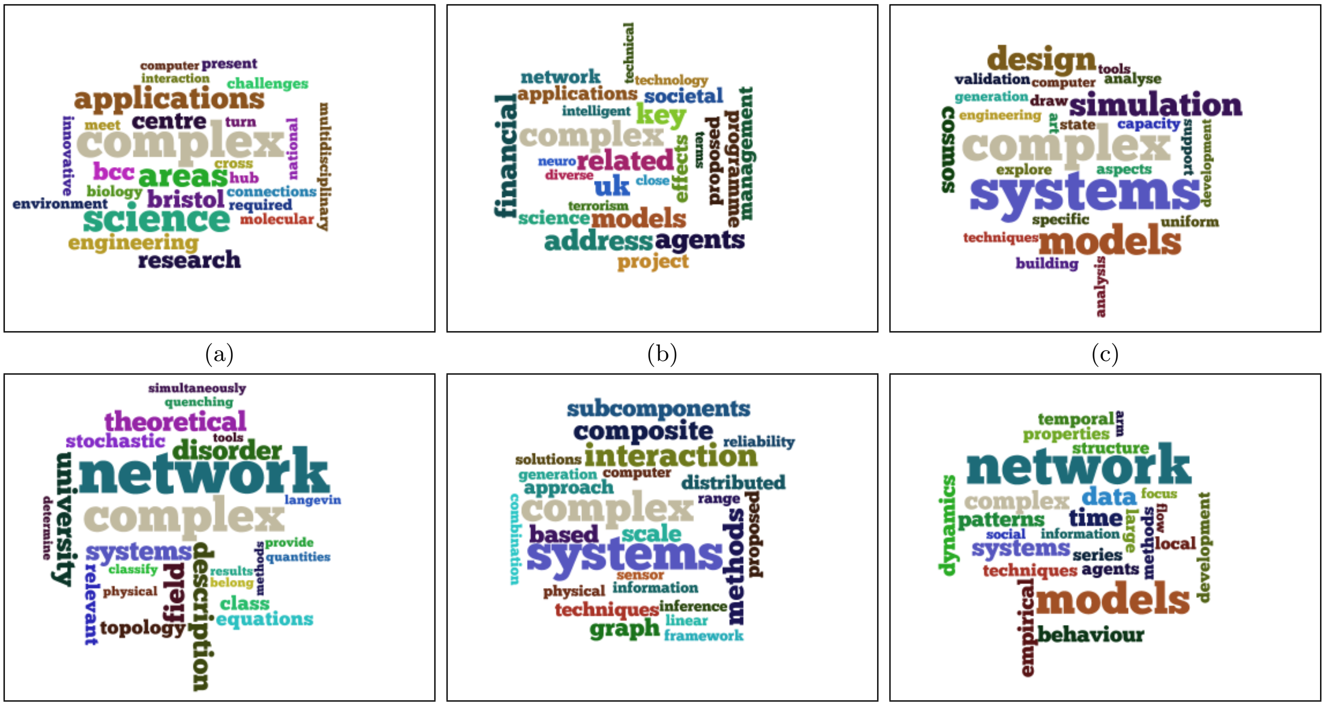
(c) Information Communication and Technology

(d) Physical Sciences

(e) Complexity

(f) Mathematical Sciences

Figure 2: A word storm describing the grants funded by six EPSRC Scientific Programs. Each cloud represents the set of all grant abstracts in the respective program.



(a)

(b)

(c)

(d)

(e)

(f)

Figure 3: A word storm containing six randomly sampled grants from the Complexity Programme (Cloud (e) in Figure 2). The word "complex", which only appeared in one cloud in Figure 2, appears in all clouds in this figure. This word conveys little information here, in contrast to the previous figure, so in this figure the word is more transparent.

Algorithm 1 Spiral Algorithm

```
Require: Words $W$, optionally positions $\mathbf{p} = \{p_w\}_{w \in W}$
Ensure: Final positions $\mathbf{p} = \{p_w\}_{w \in W}$
for all words $w \in \{w_1, \ldots, w_M\}$ do
    if initial position $p_w$ unsupplied, sample from Gaussian
    count ← 0
    while $p_w$ not valid $\wedge$ count < Max Iteration do
        Move $p_w$ one step along a spiral path
        count ← count + 1
    end while
    if $p_w$ not valid then
        Restart with a larger frame
    end if
end for
```

Algorithm 2 Iterative Layout Algorithm

```
Require: Storm $v_i = (W_i, \{c_{iw}\}, \{s_{iw}\})$ without positions
Ensure: Word storm $\{v_1, \ldots, v_N\}$ with positions
for $i \in \{1, \ldots, N\}$ do
    $\mathbf{p}_i \leftarrow$ SpiralAlgorithm($W_i$)
end for
while Not Converged $\wedge$ count < Max Iteration do
    for $i \in \{1, \ldots, N\}$ do
        $p'_{iw} \leftarrow \frac{1}{|\mathcal{V}_w|} \sum_{v_j \in \mathcal{V}_w} p_{jw}, \quad \forall w \in W_i$
        $\mathbf{p}_i \leftarrow$ SpiralAlgorithm($W_i$, $\mathbf{p}'_i$)
    end for
    count = count + 1
end while
```

4. CREATING A STORM

In this section, we present novel algorithms to build a storm. The simplest method would of course be to simply run the single-cloud algorithm of Section 3 independently for each document, but the resulting storms would typically violate the principle of coordination of similarity (Section 2.3) because words will tend to have different colors, orientations, and layouts even when they are shared between documents. Instead, our algorithms will coordinate the layout of different clouds, so that when words appear in more than one cloud, they have the same color, orientation, and position. In this way, if the viewer finds a word in one of the clouds, it is easy to check if it appears in any other clouds.

We represent each document as a vector u_i, where u_{iw} is the count of word w in document i. A word cloud v_i is a tuple $v_i = (W_i, \{p_{iw}\}, \{c_{iw}\}, \{s_{iw}\})$, where W_i is the set of words that are to be displayed in cloud i, and for any word $w \in W_i$, we define $p_{iw} = (x_{iw}, y_{iw})$ as the position of w in the cloud v_i, c_{iw} the color, and s_{iw} the font size. We write $\mathbf{p}_i = \{p_{iw} \,|\, w \in W_i\}$ for the set of all word locations in v_i.

Our algorithms will focus on coordinating word locations and attributes of words that are shared in multiple clouds in a storm. However, it is also possible to select the words that are displayed in each cloud in a coordinated way that considers the entire corpus. For example, instead of selecting words by their frequency in the current document, we could use global measures, such as $tf * idf$, that could emphasize the differences among clouds and would deal naturally with stop words. In preliminary experiments, however, we subjectively preferred storms produced using tf alone.

4.1 Coordinated Attribute Selection

A simple way to improve the coordination of a storm is to ensure that words that appear in more than one cloud are displayed with the same color and orientation across clouds. We can go a bit farther than this, however, by encoding information in the words' color and orientation. In our case, we decided to use color as an additional way of encoding the relevance of a term in the document. Rather than encoding this information in the hue, which would require a model of color saliency, instead we control the color transparency. We choose the alpha channel of the color to correspond to the inverse document frequency *idf* of the word in the corpus. In this way, words that appear in a small number of documents will have opaque colors, while words that occur in many documents will be more transparent. In this way the color choice emphasizes differences among the documents, by making more informative words more noticeable.

4.2 Coordinated Layout: Iterative Algorithm

Coordinating the positions of shared words is much more difficult than coordinating the visual attributes. In this section we present the first of three algorithms for coordinating word positions. In the same manner that we have set the color and the orientation, we want to set the position $p_{wi} = p_{wj} \; \forall v_i, v_j \in \mathcal{V}_w$, where $\mathcal{V}_w$ is the set of clouds that contain word w. The task is more challenging because it adds an additional constraint to the layout algorithm. Instead of only avoiding overlaps, now we have the constraint of placing the words in the same position across the clouds. In order to do so, we present a layout algorithm that iteratively generates valid word clouds changing the location of the shared words to make them converge to the same position in all clouds. We will refer to this procedure as the iterative layout algorithm, which is shown in Algorithm 2.

In particular, the iterative layout algorithm works by repeatedly calling the spiral algorithm (Section 3) with different desired locations for the shared words. At the first iteration, the desired locations are set randomly. Subsequently, the new desired locations are chosen by averaging the previous final locations of the word in the different clouds. That is, the new desired location for word w is $p'_w = |\mathcal{V}_w|^{-1} \sum_{v_j \in \mathcal{V}_w} p_{wj}$. Thus, the new desired location for word w is the same for all clouds $v_j \in \mathcal{V}_w$. Changing the locations of shared words might introduce new overlaps, so we run the spiral algorithm again to remove any overlaps.

Ideally this process would be iterated until the word locations converge, that is, when the spiral algorithm does not modify the given positions. At that point all shared words will be in precisely identical positions across the clouds. However, this process does not always converge, so in practice, we stop after a fixed number of iterations.

In practice we find a serious problem with the iterative algorithm. It tends to move words far away from the center, because this makes it easier to place shared words in the same position across clouds. This results in sparse layouts with excessive white space that are visually unappealing.

4.3 Coordinated Layout: Gradient Approach

In this section, we present a new method to build a storm by solving an optimization problem. This will provide us with additional flexibility to incorporate aesthetic constraints into storm construction, because we can incorporate them as additional terms in the objective function. This will allow us to avoid the unsightly sparse layouts which are sometimes produced by the iterative algorithm.

We call the objective function the Discrepancy Between Similarities (DBS). The DBS is a function of the set of clouds $v_{1:N} = \{v_1, \ldots, v_N\}$ and the set of documents $u_{1:N} = \{u_1, \ldots, u_N\}$, and measures how well the storm fits the document corpus. It is:

$$f_{u_{1:N}}(v_{1:N}) = \sum_{1 \le i < j \le N} (d_u(u_i, u_j) - d_v(v_i, v_j))^2 + \sum_{1 \le i \le N} c(u_i, v_i),$$

where d_u is a distance metric between documents and d_v a metric between clouds. The DBS is to be minimized as a function of $\{v_i\}$. The first summand, which we call stress, formalizes the idea that similar documents should have similar clouds and different documents, different clouds. The second summand uses a function that we call the *correspondence function* $c(\cdot, \cdot)$, which should be chosen to ensure that each cloud v_i is a good representation of its document u_i.

The stress term of the objective function is inspired by multidimensional scaling (MDS), a classical method for dimensionality reduction [2]. Our use of the stress function is slightly unusual, because instead of projecting the documents onto a low-dimensional space, such as $\mathbb{R}^2$, we are mapping documents to the space of word clouds. The space of word clouds is itself high-dimensional, and indeed, might have greater dimension than the original space. Additionally, the space of word clouds is not Euclidean because of the non-overlapping constraints.

For the metric d_u among documents, we use Euclidean distance. The dissimilarity function d_v between clouds is

$$d_v(v_i, v_j) = \sum_{w \in \mathcal{W}} (s_{iw} - s_{jw})^2 + \kappa \sum_{w \in W_i \cap W_j} (x_{iw} - x_{jw})^2 + (y_{iw} - y_{jw})^2,$$

where $\kappa \geq 0$ determines the weight given to differences in font size versus differences in location. Note that the first summand considers all words in either cloud, and the second only the words that appear in both clouds. (If a word does not appear in a cloud, we treat its size as zero.) The intuition is that clouds are similar if their words have similar sizes and locations. In contrast to the previous layout algorithm, by optimizing this function also determines the words' sizes.

The difference between the objective functions for MDS and DBS is that the DBS adds the correspondence function $c(u_i, v_i)$. In MDS, the position of a single data point in the target space is not interpretable on its own, whereas in our case each word cloud must accurately represent its document. Ensuring this is the role of the correspondence function. In this work we use $c(u_i, v_i) = \sum_{w \in W_i} (u_{iw} - s_{iw})^2$, where u_{iw} is the tf of word w.

We also need to ensure that words do not overlap, and to favor compact configurations. We introduce these constraints as two penalty terms. When two words overlap, we add a penalty proportional to the square of the the minimum distance required to separate them; call this distance $O_{i;w,w'}$. We favor compactness by adding a penalty proportional to the the squared distance from each word towards the center of the image, which by convention is the origin.

Incorporating these two penalties, the final objective function is $g(v_{1:N}) = f_{u_{1:N}}(v_{1:N}) + r(v_{1:N})$, where r is

$$r(v_{1:N}) = \lambda \sum_{i=1}^{N} \sum_{w,w' \in W_i} O_{i;w,w'}^2 + \mu \sum_{i=1}^{N} \sum_{w \in W_i} ||p_{iw}||^2,$$

where λ and μ are parameters that determine the strength of the overlap and compactness penalties, respectively.

We optimize this by solving a sequence of optimization problems for increasing values $\lambda_0 < \lambda_1 < \lambda_2 < \ldots$ of the overlap penalty. We increase λ exponentially until no words overlap in the final solution. Each subproblem is minimized using gradient descent starting from the previous solution.

4.4 Coordinated Layout: Combined Algorithm

The iterative and gradient algorithms are complementary. The iterative algorithm is fast, but it does not enforce that clouds stay compact. The gradient method can create compact clouds, but requires many iterations to converge, and the layout strongly depends on the initialization. Therefore we combine the two methods, using the final result of the iterative algorithm as the starting point for the gradient method. From this initialization, the gradient method converges much faster, because it starts off without overlapping words. The gradient method tends to improve the initial layout significantly, because it pulls words closer to the center, creating a more compact layout. Also, the gradient method tends to pull together the locations of shared words for which the iterative method was not able to converge to a single position. The above steps are run only for words that appear in multiple clouds. We lay out the remaining words that are unique to single clouds at the end using the spiral algorithm. This leads to improvements in the running time, since we deal with fewer words during the first phase, and it results in more compact clouds, because the unique words, being less constrained, can fit into odd patches of whitespace. By default, all clouds in this paper are created using this combined algorithm.

5. EVALUATION

The evaluation is divided in three parts: a qualitative analysis, an automatic analysis, and a user study. We use two different data sets. First, we use the scientific papers presented in the ICML 2012 conference, where we deployed a storm on the main conference Web site to compare the presented papers and help people decide among sessions[1].

Second, we use a data set provided by the Research Perspectives project[2] [10], which aims to offer a visualization of the research portfolios of funding agencies. The data contains 2358 abstracts of funded research grants from the UK's Engineering and Physical Sciences Research Council (EPSRC). Each grant belongs to exactly one of the following programmes: Information and Communications Technology (626 grants), Physical Sciences (533), Mathematical Sciences (331), Engineering (317), User-Led Research (291) and Materials, Mechanical and Medical Engineering (264).

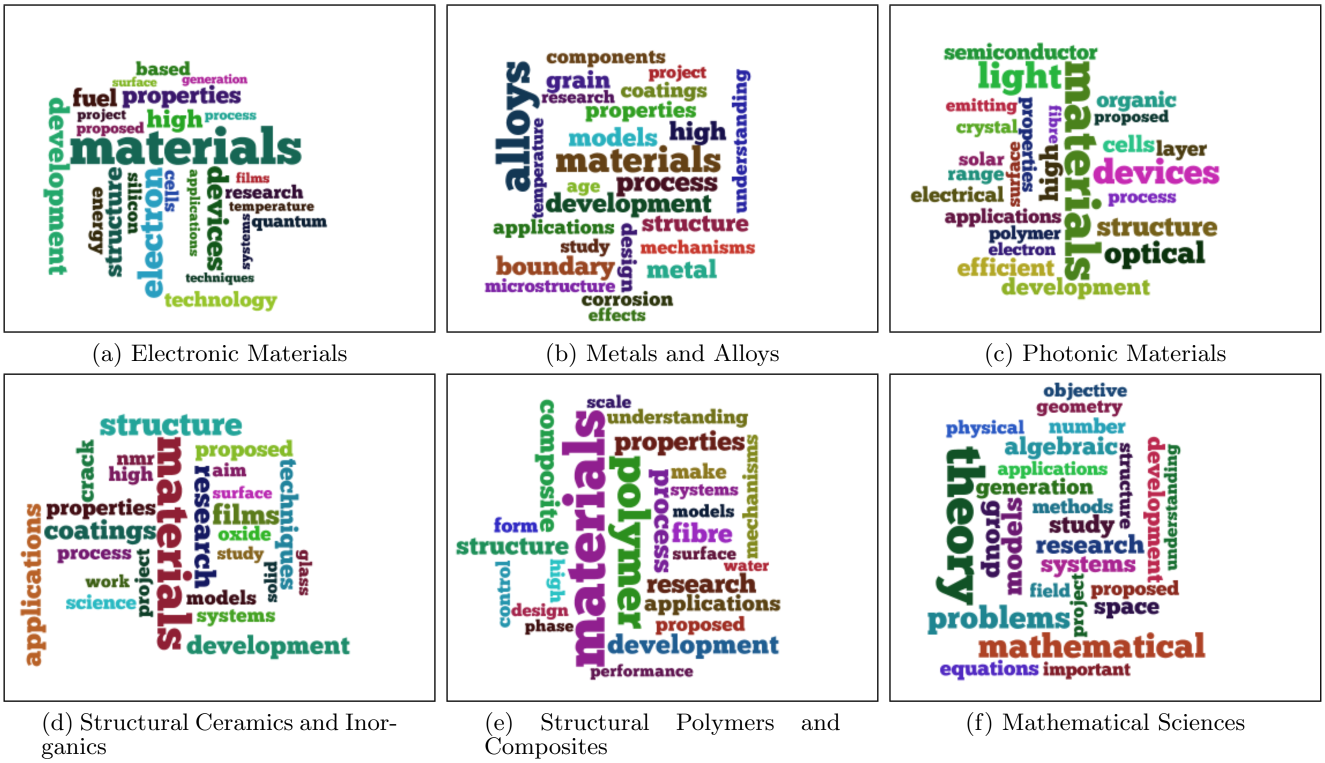
5.1 Qualitative Analysis

This section discusses coordinated word storms in qualitative fashion, describing the additional information about a corpus that they make apparent.

First, we consider a storm that displays six research programmes from EPSRC programmes, five of which are different subprogrammes of material sciences and the sixth one is the mathematical sciences programme. For this data set we present both a set of independent clouds (Figure 4) and

[1] `http://icml.cc/2012/whatson/`

[2] Also see `http://www.researchperspectives.org`



(a) Electronic Materials

(b) Metals and Alloys

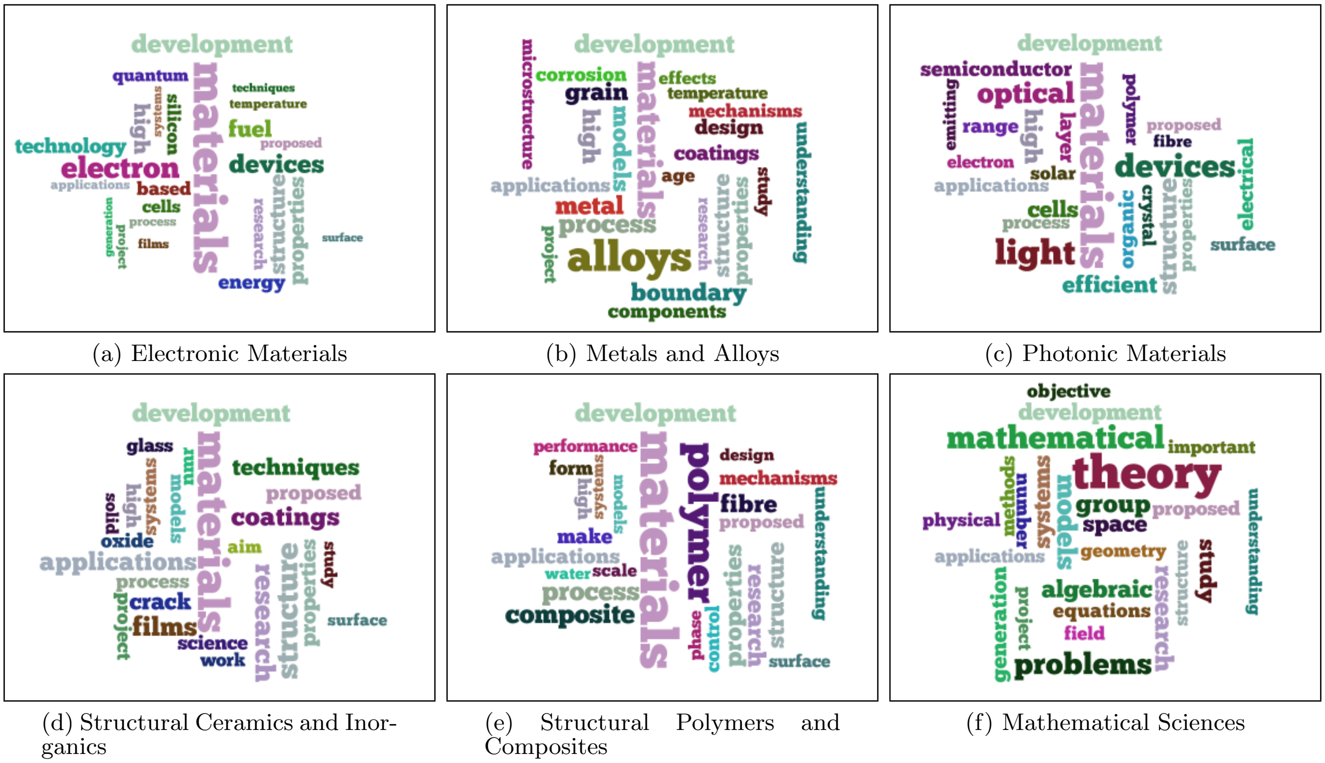
(c) Photonic Materials

(d) Structural Ceramics and Inorganics

(e) Structural Polymers and Composites

(f) Mathematical Sciences

Figure 4: Independent Clouds representing six EPSRC Scientific Programmes. These programmes are also represented as a coordinated storm in Figure 5.



(a) Electronic Materials

(b) Metals and Alloys

(c) Photonic Materials

(d) Structural Ceramics and Inorganics

(e) Structural Polymers and Composites

(f) Mathematical Sciences

Figure 5: Coordinated storm representing six EPSRC Scientific Programmes. These programmes are also represented as independent clouds in Figure 4. Compared to that figure, here it is much easier to see the differences between clouds.

	Time (s)	Compactness (%)	Accuracy (%)
Lower Bound	-	-	26.5 ± 3.9
Independent Clouds	143.3	35.12	23.4 ± 3.8
Coordinated Storm (Iterative)	250.9	20.39	54.7 ± 4.5
Coordinated Storm (Combined)	2658.5	33.71	54.2 ± 4.5
Upper Bound	-	-	67.9 ± 4.2

Table 1: Automatic evaluation of word storm algorithms. The small numbers indicate 95% confidence intervals.

a storm generated by the combined algorithm (Figure 5). From either set of clouds, we can get a superficial idea of the corpus. We can see the most important words such as "materials", which appears in the first five clouds, and some other words like "alloys", "polymer" and "mathematical". However, it is hard to get more information than this from the independent clouds.

On the other hand, by looking at the coordinated storm we can obtain more information. First, it is instantly clear that the first five documents are similar and that the sixth one is different from the others. This is because the storm reveals the shared structure in the documents, formed by shared words such as "materials", "properties" and "applications". Second, we can easily tell the presence or absence of words across clouds because of the consistent attributes and locations. For example, we can quickly see that "properties" does not appear in the sixth cloud or that "coatings" only occurs in two of the six. Finally, the transparency of the words allows us to spot the informative terms quickly, such as "electron" (a), "metal" (b), "light" (c), "crack" (d), "composite" (e) and "problems" (f). All of these terms are informative of the document content but are difficult to spot in the independent clouds of Figure 4. Overall, the coordinated storm seems to afford deeper analysis than the independently generated clouds.

Similarly, from the ICML 2012 data set, Figure 1 shows a storm containing all the papers from a single conference session. It is immediately apparent from the clouds that the session discusses optimization algorithms. It is also clear that the papers (c) and (d) are very related since they share a lot of words such as "sgd", "stochastic" and "convex" which results in two similar layouts. The fact that shared words take similar positions can force unique words into similar positions as well, which can make it easy to find terms that differentiate the clouds. For example, we can see how "herding" (f), "coordinated" (g) and "similarity" (h) are in the same location or "semidefinite" (a), "quasi-newton" (b) and "nonsmooth" (d) are in the same location.

Finally, Figures 2 and 3 show an example of a hierarchical set of storms generated from the EPSRC grant abstracts. Figure 2 presents a storm created by grouping all abstracts by their top level scientific program. There we can see two pairs of similar programmes: Chemistry and Physical Sciences; and Engineering and Information Communication and Technology. In Figure 3, we show a second storm composed of six individual grants from the Complexity programme (Cloud (e) in Figure 2). It is interesting to see how big words in the top level such as "complex", "systems", "network" and "models" appear with different weights in the grant level. In particular, the term "complex", that it is rare when looking at the top level, appears everywhere inside the complexity programme. Because of our use of transparency, this term is therefore prominent in the top level storm but less noticeable in the lower level storm.

5.2 Automatic Evaluation

We propose a novel automatic method to evaluate word storm algorithms. The objective is to assess how well the relations among documents are represented in the clouds. The motivation is similar in spirit to the celebrated BLEU measure in machine translation [12]: Automatic evaluation, rather than a user study, allows rapid and inexpensive comparison of algorithms. Our automatic evaluation requires a corpus of labelled documents, e.g., with a class label that indicates their topic. The main idea is: If the visualization is faithful to the documents, then it should be possible to classify the documents using the pixels in the visualization rather than the words in the documents. So we use classification accuracy as a proxy measure for visualization fidelity.

In the context of word storms, the automatic evaluation consists of: (a) generating a storm from a labelled corpus with one cloud per document, (b) training a document classifier using the pixels of the clouds as attributes and (c) testing the classifier on a held out set to obtain the classification accuracy. More faithful visualizations are expected to have better classification accuracy.

We use the Research Perspectives EPSRC data set with the research programme as the class label. Thus, we have a single-label classification problem with 6 classes. The data was randomly split into a training and test set using an 80/20 split. We use the word storm algorithms to create one cloud per abstract, so there are 2358 clouds in total. We compare three layout algorithms: (a) creating the clouds independently using the spiral algorithm, which is our baseline; (b) the iterative algorithm with 5 iterations and (c) the combined algorithm, using 5 iterations of the iterative algorithm to initialize the gradient method.

We represent each cloud by a vector of the RGB values of its pixels. We perform feature selection, discarding features with zero information gain. We classify the clouds by using support vector machines with normalized quadratic kernel and an all-pairs method. As a lower bound, classifying all instances as the most common class (ICT) yields an accuracy of 26.5%. To obtain an upper bound, we classify the documents directly using bag-of-words features from the text, which should perform better than transforming the text into a visualization. Using a support vector machine with normalized quadratic kernel and an all-pairs method, this yields an accuracy of 67.9%.

Apart from the classification accuracy, we also report the running time of the layout algorithm (in seconds),[3] and the compactness of the word clouds. We use this measure be-

[3] All experiments were run on a 3.1 GHz Intel Core i5 server with 8GB of RAM.

cause informally we noticed that more compact clouds tend to be more visually appealing. We compute the compactness by taking the minimum bounding box of the cloud and calculating the percentage of non-background pixels.

The results are shown in Table 1. Creating the clouds independently is faster than any coordinated algorithm and also produces very compact clouds. However, for classification, this method is no better than random. The algorithms to create coordinated clouds, the iterative and the combined algorithm, each achieve a 54% classification accuracy, which is significantly higher than the lower bound. This confirms the intuition that by coordinating the clouds, the relations among documents are better represented. We also report the 95% confidence intervals for the accuracy, which indicate that the difference in accuracy between either of the coordinated storm methods and the independent word cloud method is statistically significant. The difference in accuracy between the two coordinated methods is not significant.

It is worth noting that this is a difficult classification problem even given the textual features. We speculate that this may be due to the degree of textual overlap between the abstracts in the different research programmes. It is less surprising that the accuracy of the classifier using independent clouds is low, because for these clouds, the color values of individual pixels have no semantics individually.

The differences between the coordinated methods can be seen in the running time and in the compactness. Although the iterative algorithm achieves much better classification accuracy than the baseline, it produces much less compact clouds. The combined algorithm, on the other hand, matches both the compactness of independently built clouds (33.71% combined and 35.12% independent) and the classification accuracy of the iterative algorithm. The combined algorithm is significantly more expensive in computation time, although this is still only 1.1s for each cloud in the storm. Therefore, although the combined algorithm requires more time, it seems the best option, because the resulting storm offers good classification accuracy without losing compactness.

A potential pitfall with automatic evaluations is that algorithms can game the system, producing visualizations that score better but look worse. It is arguable that this may have happened in machine translation, in which BLEU has been optimized by the research community for many years. We attempt to avoid this by choosing an measure for the automatic evaluation (classification accuracy) that is not directly optimized by the algorithms. But the concern of "research community overfitting" could become more serious if automated evaluation of visualization is widely adopted.

5.3 User Study

In order to confirm our results using the automatic evaluation, we conducted a pilot user study comparing the standard independent word clouds with coordinated storms created by the combined algorithm. The study consisted of 5 multiple choice questions. In each of them, the users were presented with six clouds and were asked to perform a simple task. The tasks were of two kinds: checking the presence of words and comparing documents. The clouds for each question were generated either as independent clouds or a coordinated storm. In every question, the user received one of the two versions randomly.[4] Although users were told in the beginning that word clouds had been built using different methods, the number of different methods was not revealed, the characteristics of the methods were not explained and they did not know which method was used for each question. Moreover, in order to reduce the effect of possible bias factors, the tasks were presented in a random order and the 6 clouds in each question were also sorted randomly. The study was taken by 20 people, so each question was answered 10 times using the independent clouds and 10 times using a coordinated storm.

Table 2 presents the results of the study. The first three questions asked the users to select the clouds that contained or lacked certain words. We manually chose words that were prominent in both sets of clouds. We report the mean precision and recall across users, as well as the time required for users to answer. The results show that although the precision and recall are high in both cases and the differences are small, the coordinated storm always has a higher score than the independent clouds. Although this result is not statistically significant, it is still remarkable because we did not explain the concept of coordinated word storm to the users, so they would not have known to look for the same words in the same locations. This might be because the structured layout helped the users to find words, even though the users did not know how the storms were laid out.

The last two questions asked the users to compare the documents and to select "the cloud that is most different from all the others" and "the most similar pair of clouds". As coordinated word storms are designed to highlight similarities and differences between documents, these are the questions in which we expect to see the greatest difference between methods. For question 4, the clouds had a cosine similarity[5] lower than 0.3 with all the others, while all others pairs had a similarity higher than 0.5. In the last question, the most similar pair of clouds had a cosine similarity of 0.71, while the score of the second most similar was 0.48. As these questions have exactly one correct answer, we report the accuracy, instead of the precision and recall.

The results for the last two questions show that the coordinated storm is much more effective than independent clouds in allowing users to compare and contrast documents. Of the users presented with the coordinated storms, 90% answered question 4 correctly and 70% answered question 5 correctly, whereas only 30% and 10% of users, respectively, answered correctly when shown the independent version. This difference is highly significant (t-test; $p < 0.005$). This confirms that coordinated storms allow the users to contrast the clouds and understand their relations, while independent clouds are misleading in these tasks.

Although the sample size is small, results favour the coordinated storm. In particular, when the users are asked to compare clouds, the differences in user accuracy are extremely large. Regarding the answering time, the differences between the two conditions are not significant.

6. RELATED WORK

Word clouds were inspired by tag clouds, which were introduced to summarize and browse a user-specified folksonomy.

[4] The random process ensured that we would have the same number of answers for each method.

[5] The documents were taken using the bag of words representation with frequencies. The cosine similarity was computed twice: considering all words in the document and only considering the top 25 words included in the cloud.

	Question		Independent clouds	Coordinated Storm
1	Select clouds with the word “technology”	Precision (%)	90	100
		Recall (%)	65	85
		Time (s)	51 ± 23	36 ± 10
2	Select clouds without the word “energy”	Precision (%)	90	93
		Recall (%)	85	95
		Time (s)	56 ± 18	40 ± 14
3	Select clouds with the words “models”, “network” and “system”	Precision (%)	75	90
		Recall (%)	90	100
		Time (s)	87 ± 35	124 ± 46
4	Select the single most unusual cloud	Accuracy (%)	30	90 *
		Time (s)	36 ± 12	23 ± 10
5	Select the most similar pair of clouds	Accuracy (%)	10	70 *
		Time (s)	54 ± 23	75 ± 19

Table 2: Results of the user study. Users are more effective at comparing documents when shown coordinated storms. The “time” rows report the mean and standard deviation across users. Stars indicate statistical significance ($p < 0.005$).

Originally, the tags were organized in horizontal lines and sorted by alphabetical order, a layout that is still used in many websites such as Flickr and Delicious. Word clouds, such as those generated by Wordle [8, 19], extend this idea to document visualization. However, the topic of visualizing corpora using word clouds has received much less attention. Researchers have proposed creating the clouds using different importance measures, such as $tf{*}idf$ [9] or by the relative frequency when only the relations of a single document have to be analysed [14, 4]. Nevertheless, it can still be difficult to compare the resulting clouds and find shared words.

Collins et al. [5] presented Parallel Tag Clouds, a method that aims to make comparisons easier by representing the documents as lists. Although alphabetical lists are informative and easy to understand, our work aims to retain the aesthetic appeal of word clouds while improving their informativeness. The closest work to ours is Cui et al. [6], which was later improved by Wu et al. [20]. This work proposes using a sequence of word clouds along with a trend chart to show the evolution of a corpus over time. They present a new layout algorithm with the goal of keeping semantically similar words close to each other in each cloud. This is a different goal from ours: Preserving semantic relations between words within a cloud is different than coordinating similarities across clouds, and does not necessarily result in similar documents being represented by similar clouds.

Our approach, in common with Wordle and many other text visualization methods, does not attempt to resolve cases of synonymy and polysemy. One can imagine potential extensions to our method to handle this, e.g., by incorporating word sense disambiguation [11] to visually distinguish different word senses, or by incorporating topic modelling methods [1] to visually conflate words that are semantically similar. These extensions are left to future work.

An alternative general approach to visualization of document collections is to employ dimensionality reduction methods, for which there is an extensive literature [7, 1, 13, 15, 18]. These methods can be used to assign each document to a single point in a low-dimensional latent space that can be explored visually. Indeed, topic models have previously been applied to the grant proposals data set [10]. Word clouds and their ilk take an alternative approach. Instead of mapping documents into a low-dimensional space, documents are mapped into a *high dimensional* space, but one that is well suited to the human visual system. One advantage of this is that the high dimensional representation, e.g., the word cloud, can convey some information about the document on its own. Another advantage is aesthetic. Word clouds are extremely popular among users — to give a crude indication, the query “word cloud” currently returns 357 million hits on Google, whereas “latent Dirichlet allocation” returns 157 thousand. This indicates the potential value of work that aims to increase the statistical informativeness of popular visualization methods that have clear aesthetic appeal.

7. CONCLUSIONS

We have introduced the concept of word storms, which is a group of word clouds designed for the visualization of a corpus of documents. We presented a series of principles for effective storms, arguing that the clouds in a storm should be built in a coordinated fashion, so that similar documents have similar clouds. We presented a novel algorithm that builds coordinated word storms, placing shared words in a similar location across clouds. Using both an automatic evaluation and a user study, we showed that coordinated storms were markedly superior to independent word clouds for comparing and contrasting documents. Future work could explore ways of organizing hierarchical storms for large collections for which it is impossible to view all of the clouds at once, and informative ways of arranging the clouds within a storm. Source code implementing the algorithms in this paper is available [3].

8. ACKNOWLEDGMENTS

We thank Mike Chantler and Fraser Halley for kindly providing access to the research grant data. This work was supported by the Engineering and Physical Sciences Research Council [grant number EP/J00104X/1].

References

[1] D. Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

[2] I. Borg and P. Groenen. *Modern Multidimensional Scaling: Theory and Applications.* Springer, 2005.

[3] Q. Castella and C. Sutton. Word storms. URL `http://groups.inf.ed.ac.uk/cup/wordstorm/wordstorm.html`.

[4] J. Clark. Clustered word clouds - Neoformix, April 2008. URL `http://www.neoformix.com/`.

[5] C. Collins, F. B. Viegas, and M. Wattenberg. Parallel tag clouds to explore and analyze faceted text corpora. pages 91–98, Oct. 2009.

[6] W. Cui, Y. Wu, S. Liu, F. Wei, M. Zhou, and H. Qu. Context-preserving, dynamic word cloud visualization. *IEEE Computer Graphics and Applications*, 30:42–53, 2010.

[7] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science (JASIS)*, 41(6): 391–407, 1990.

[8] J. Feinberg. Wordle. In J. Steele and N. Iliinsky, editors, *Beautiful Visualization Looking at Data through the Eyes of Experts*, chapter 3. O'Reilly Media, 2010.

[9] Y. Hassan-Montero and V. Herrero-Solana. Improving tag-clouds as visual information retrieval interfaces. In *InScit2006: International Conference on Multidisciplinary Information Sciences and Technologies*, 2006.

[10] O. Khalifa, D. Corne, M. Chantler, and F. Halley. Multi-objective topic modelling. In F. F. Purshouse and S. Greco, editors, *Evolutionary Multi-Criterion Optimization (EMO)*, 2013.

[11] R. Mihalcea. Word sense disambiguation. In *Encyclopedia of Machine Learning*. Springer, 2007.

[12] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. BLEU: a method for automatic evaluation of machine translation. In *Annual Meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[13] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290: 2323–2326, 2000.

[14] J. Steele and N. Iliinsky. *Beautiful Visualization: Looking at Data through the Eyes of Experts*. O'Reilly & Associates Inc, 2010. ISBN 1449379869.

[15] J. B. Tenenbaum, V. De Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500): 2319–2323, 2000.

[16] E. R. Tufte. *Visual Explanations: Images and Quantities, Evidence and Narrative*. Graphics Press LLC, 1997.

[17] E. R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press LLC, 2nd edition, 2001.

[18] L. Van der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9 (2579-2605):85, 2008.

[19] F. B. Viegas, M. Wattenberg, and J. Feinberg. Participatory visualization with Wordle. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1137–1144, Nov. 2009.

[20] Y. Wu, T. Provan, F. Wei, S. Liu, and K.-L. Ma. Semantic-preserving word clouds by seam carving. *Comput. Graph. Forum*, 30(3):741–750, 2011.