DocuScope Assessment

Validating DocuScope rhetorical tags in the CITF email corpus

Created November 12, 2026
Updated November 16, 2026

Table of contents

1 Purpose **2**

2 Starting point **2**

3 Distribution of rhetorical tags **2**

4 Detecting likely tagging errors **3**
- 4.1 Excluding automated notifications . . . 4
- 4.2 Zones . . . 4
- 4.3 Checking the zones . . . 8
- 4.4 Error rules . . . 9
- 4.5 Issues that span more than one tag . . . 10

5 The coding sample and coding tool **11**
- 5.1 Rationale . . . 11
- 5.2 Stratification . . . 11
- 5.3 The coding tool . . . 12

6 Preliminary comparison across tag versions **12**
- 6.1 Category shares . . . 12
- 6.2 Phrases . . . 13
- 6.3 Sender groups . . . 13

7 Validating rules and correcting tags (planned) **13**

8 Planned analysis **14**

9 Reproducibility **15**

10 Next steps **16**

References **16**

1 Purpose

[1] The CITF email project studies research data sharing through the correspondence that carried it. A first close reading of 30 emails showed that how people wrote mattered as much as what they wrote about, and we adopted DocuScope, following Brown and Laudenbach (2022), to capture the *how* at corpus scale. The project uses DocuScope for rhetorical categories only.

[2] This session asked whether DocuScope's rhetorical tags describe these emails accurately enough to support that use, and built a reproducible procedure to find tagging errors, measure them, and correct the ones that can be corrected. Every step that changes the data is code with fixed parameters, and every human judgement is stored as a coded file, so that each result can be rerun and audited.

2 Starting point

[3] Melissa's notebook, since split into `corpus.ipynb`, `rhetoric.ipynb` and `docuscope.ipynb`, sets up everything this session relies on. It loads `msgs10` and `recip4`, flattens and deduplicates the emails, keeps the 10,397 emails linked to a CITF study ID, tags email bodies with the `en_docusco_spacy` model (version 1.5), and exports 30 tagged emails to `docuscope_sample_30.html`. Block numbers without a notebook name refer to `docuscope.ipynb`.

[4] `corpus.ipynb` block 3 defines the pool that this session works from: study emails of 300 to 2,000 characters, with emails that look like they contain credentials removed. The pool holds 6,086 emails.

[5] Block 2 sorts the 19 rhetorical categories into six working groups (`rhetoric_groups`), which made the 30-email sample easier to read. This session works with the categories directly, since we plan to derive groups from the data through a factor analysis once the tags are cleaned (Section 8).

[6] The notebooks run on the CITF analysis server, reached from VSCodium through the Open Remote SSH extension, in the environment `~/citf-emails-env`, built from the repository's `requirements.txt`. Section 9 lists the software versions.

3 Distribution of rhetorical tags

[7] `rhetoric.ipynb` block 1 parses all 6,086 emails in the pool, and its block 2 keeps only rhetorical tags, counting tag spans (`doc.ents`), the same unit as the 30-email sample in block 2.

[8] The parse produced 374,940 tags, of which 176,836 (47.2%) were rhetorical tags. Narrative (35,341), Positive (30,102) and Interactive (28,226) together carried

more than half of them. The median email held 25 rhetorical tags, and rhetorical tags made up a median 48% of each email's tags.

[9] The 15 most frequent phrases in each category, which cover 53.5% of rhetorical tags, showed that the raw counts mix rhetoric with text no sender composed and with words DocuScope reads in a sense other than the one their sentence gives them. Within those phrases, at least 9,979 tags (5.6% of rhetorical tags) came from the CITF name, external-sender banners, confidentiality footers, French text and image markup, and 5,929 (3.4%) were domain nouns such as agreement, consent, health, assay and research. These figures cover only the visible half of the tags, so they motivated a systematic check and are not themselves results.

4 Detecting likely tagging errors

[10] Our strategy is to clean the tags with error rules. Each DocuScope category stands for a rhetorical function, such as hedging a claim or addressing the reader. A tag is inaccurate when the word or phrase it marks does not perform that function in its sentence, as when "May" in "May 8th" carries a ConfidenceHedged tag although it names a month and qualifies no claim. An error rule describes, in code, one recurring form of inaccurate tag. The inaccuracies in Section 3 recur in recognizable forms across thousands of emails, so a rule can find every instance of its form, the same way on every run, in terms anyone can read and question. Human judgement then goes into testing the rules and estimating the inaccuracies that no rule catches. If those remaining inaccuracies prove substantial, a supervised model trained on human judgements becomes the next step. In the human review, described in Section 5, two coders independently judge a sample of tags. Block 5 then measures agreement between the coders, estimates each rule's hit rate and each category's remaining inaccuracy with confidence intervals and stratum weights, keeps the rules that meet thresholds set in advance, and relabels their matches with a log of every change.

[11] The work spans four blocks. `corpus.ipynb` block 5 and block 3 leave out automated notifications (Section 4.1), `corpus.ipynb` blocks 4 and 5 divide every email into zones (Section 4.2), and `zoning.ipynb` draws the emails for checking the zones by hand (Section 4.3). Block 3 gives each tag the zone of the line it falls on, flags rhetorical tags in authored text that match an error rule (Section 4.4), and lists inaccuracies that span more than one tag (Section 4.5). It saves every rhetorical tag with its zone and rule matches to `rhetoric_tags.parquet`, and then draws the coding sample for the human review (Section 5). All patterns, meaning the regular expressions and word lists that define reply headers, banners, closing lines, names, domain nouns and each error rule, sit at the top of the cell that uses them, so that each can be inspected and changed.

4.1 Excluding automated notifications

[12] Automated notifications have no author, so `corpus.ipynb` block 5 and block 3 leave them out by sender. The run excluded 30 notifications from DocuSign, SharePoint and WeTransfer, leaving 6,056 emails and 174,765 rhetorical tags. The count is small because the pool's filters already remove most notifications, and because the rule identifies automated emails by sender address alone. Automated messages sent from a person's address therefore remain in the pool, including ticket confirmations ("Reference#: REF00000 Please use this number when inquiring") and out-of-office replies. Catching them requires patterns on email content, such as "out of the office", "absent du bureau" or "Reference#", which no block includes yet.

4.2 Zones

[13] An email body holds more than what its sender wrote for that message. It also carries earlier messages quoted below a reply, notices that mail systems insert, markup, signature blocks and boilerplate lines that recur across many emails. DocuScope tags all of this text alike, but tags in it describe other writers, mail software or standing institutional text, not the sender's rhetoric. Because the same signatures and notices repeat word for word across hundreds of emails, their tags also make senders look more alike than their writing is. `corpus.ipynb` blocks 4 and 5 therefore divide each email into zones and keep only the authored zone for analysis.

[14] `corpus.ipynb` block 4 labels the email bodies with Chipmunk (Bevendorff et al. 2020), a neural model trained to segment emails. Chipmunk gives each line of an email body one of 15 labels, such as paragraph, closing, personal signature, quotation and mail client signature. The model runs on the CITF analysis server, so no email text leaves it.

[15] We chose Chipmunk after comparing the published tools for email zoning, the task of dividing an email into parts such as greeting, body, signature and quoted text (Lampert, Dale, and Paris 2009). Rule-based libraries such as talon, email_reply_parser and mail-parser-reply detect only quotations and signatures, and their authors report no evaluation. Zebra (Lampert, Dale, and Paris 2009) exists only in third-party reimplementations, and Chipmunk outperformed Quagga (Repke and Krestel 2018) in its authors' tests. Chipmunk is peer-reviewed, its authors released its code, training data and trained model, and it labels greetings and closings and separates personal signatures from notices that mail programs add.

[16] We tested Chipmunk extensively before using it. Its authors trained it on Gmane, an archive of public mailing lists. They tested it on Gmane and on Enron, a corpus of corporate email. We reran both tests and reproduced their results (Zoning Tests). We also ran Chipmunk on five CITF emails and compared its zones with a provisional labelling made during testing, which has not been reviewed.

[17] We compared Chipmunk with Okapi (Jardim, Rei, and Almeida 2021). Okapi is a multilingual extension of Chipmunk that keeps its 15 zones and Gmane training emails. Chipmunk reads the layout of each line and the lines around it. Okapi reads the wording of each line through a multilingual language model. Okapi's authors published its design but no trained model. We therefore built Okapi from their paper and trained it ourselves, and the Okapi results in the table come from our rebuild. The two models score alike on Gmane and Enron. On the five CITF emails Okapi labelled whole signature blocks as authored text in three of them and Chipmunk did not. The table gives the share of lines each model labels correctly. It also gives the results that Chipmunk's authors report for Quagga and an SVM baseline on the same benchmarks.

Model	Gmane, published	Gmane, our run	Enron, published	Enron, our run	Five CITF emails (provisional)
Chipmunk	96%	95.5%	88%	88.8%	54 of 66 lines
Okapi	96%	96.4%	88%	85.3%	29 of 66 lines
Quagga	94%	not run	83%	not run	not run
SVM baseline	80%	not run	72%	not run	not run

[18] The table shows that Chipmunk scores highest on Enron and on the five CITF emails. Our runs used the authors' own evaluation sets, namely their 300 Gmane validation emails, on which they selected the released model, and 296 of their 300 Enron emails.

[19] The tests also exposed one weakness, which the code in `corpus.ipynb` block 5 corrects. On Enron, Chipmunk and Okapi both read about a quarter of banner and disclaimer lines as authored text. That block therefore adds the disclaimer rule and the banner regular expressions described below.

[20] `corpus.ipynb` block 5 maps Chipmunk's 15 line labels onto six zones. Chipmunk was trained on technical mailing lists, so most of its labels describe content such as code patches, log output and tables. Our analysis keeps only authored text, so the block merges the labels into one authored zone and four zones for text the sender did not write. It adds a sixth zone, markup, matched by regular expressions. We keep the four non-authored zones separate to see how many of each category's tags come from signatures, banners, quoted messages and other text. Block 3, which flags likely tagging errors, reports these counts.

Zone	What it covers	Chipmunk labels
Authored	Text the sender wrote for this message, including greetings and closing lines	paragraph, salutation, closing
Signature	Signature blocks: names, titles, organizations, addresses	personal_signature
Banner	Inserted notices: external-sender warnings, confidentiality disclaimers, work-from-home alerts, land acknowledgements	mua_signature
Quoted	Earlier messages below a reply or forward header	quotation, quotation_marker, inline_headers
Markup	Angle-bracket tags, image references, "image removed by sender", URLs, Outlook Web Access link labels	none; regular expressions only
Other	Separators, tables, attachment lists, meeting-invitation details, code and log lines	all other labels; attachment and meeting-invitation lines by regular expressions

[21] To check Chipmunk across the whole corpus, `corpus.ipynb` block 5 compares Chipmunk's zones with zones assigned by regular expressions. The regular expressions mark reply and forward headers, external-sender banners, closing lines and markup. They place the signature after the last closing line and treat lines repeated across many emails as template text. The two methods agree on 73.5% of the 62,906 non-blank lines, and each catches text the other misses. Chipmunk finds signatures in emails with no closing line, confidentiality disclaimers, and attachment and dial-in lines. The regular expressions find external-sender banners, reply headers that carry no ">" marks, and markup. The block therefore combines the two.

[22] `corpus.ipynb` block 5 refines Chipmunk's zones in two stages. Chipmunk sometimes gives a line a label that clashes with the text around it, and five rules first correct these errors.

1. **Greetings and closings are authored.** Greetings and closings ("Dear Dr. Smith", "Kind regards") are part of how these emails do their work.

Chipmunk's salutation and closing labels already map to the authored zone. A closing such as "Kind regards," that Chipmunk labels signature or other also becomes authored.

2. **Disclaimers are banner.** Text that Chipmunk labels authored, signature or other but that contains disclaimer wording, such as "intended recipient" or "received this message in error", becomes banner.
3. **Organization names and job titles inside a signature block are signature.** Chipmunk sometimes labels an organization name or job title as authored when the text above and below it is signature. Such text becomes signature unless it ends a sentence.
4. **Sentences inside the message are authored.** Chipmunk sometimes labels a sentence as banner or other when the text above and below it is authored. Such a sentence becomes authored if it has four or more words and no tabs.
5. **Signature lines after the message are signature.** Chipmunk sometimes labels a whole signature as authored when no closing precedes it. Short lines without sentence punctuation at the end of the message become signature when a signature line follows them or when they hold a name with credentials, a title, an organization or contact details. A sign-off name alone, such as "Sam", stays authored.

[23] Rule 5 and the attachment and meeting-invitation lines come from the zone check (Section 4.3).

[24] Zones matched by regular expressions then take precedence, in this order: markup, attachment and meeting-invitation lines (other), banner, and quoted, which covers everything from the first reply or forward header on. A line of underscores counts as a reply header only when a "From:" or "De :" line follows it, so meeting invitations stay out of the quoted zone. The banner expressions cover English and French external-sender warnings, disclaimer headings, remote-work notices and land acknowledgements. The block prints a sample of the lines each rule changes, and saves the final zone of every line to `email_zones.parquet`. The table gives the resulting share of each category's tags in each zone.

Category	Tags	Authored (%)	Signature (%)	Banner (%)	Quoted (%)
Strategic	12,388	51.4	41.8	1.3	1.3
Negative	7,961	66.2	5.8	26.8	0.7
Positive	29,856	77.8	13.5	5.2	1.2
Interactive	27,744	79.2	5.9	12.1	1.7
Narrative	34,703	85.6	3.8	7.7	1.5

Category	Tags	Authored (%)	Signature (%)	Banner (%)	Quoted (%)
FirstPerson	9,154	95.0	2.0	1.2	1.6

[25] Strategic is the clearest case, with 41.8% of its tags in signatures. DocuScope tags "research" and "task force" as Strategic, and these were the two most frequent Strategic phrases in `rhetoric.ipynb` block 2 (3,341 and 2,694 tags). Both recur in signature blocks, in job titles such as "Research Assistant" and "Research Associate" and in the name COVID-19 Immunity Task Force.

4.3 Checking the zones

[26] The error rules apply only to tags in the authored zone, so we validated the zones in `email_zones.parquet` against zones assigned by a human coder. The coder saw neither these zones nor Chipmunk's labels. `zoning.ipynb` draws the emails and writes the labelling tool. The coder followed these rules:

- A name typed as the sign-off is authored. A name followed by a title or contact details starts the signature.
- The sender's signature placed below a quoted message is signature.
- Standing notices, such as confidentiality disclaimers, remote-work notices and land acknowledgements, are banner wherever they appear, including inside a signature.
- The quoted zone starts at the reply or forward header, or at the separator line that opens it.
- A line that holds only markup is markup, including inside a signature. A line that mixes an image and text takes the zone of the text.
- Separators that open no quoted message, attachment lists and meeting-invitation details are other.
- A line of punctuation alone takes the zone of the lines around it.

[27] We compared the coder's zones with those in `email_zones.parquet` on a first set of 80 emails and added rules to correct the differences. We drew 20 of these emails at random and 60 from emails containing banners, quoted text or a line whose zone differs from Chipmunk's label. A random sample would contain too few of these lines to test the rules. The 30 emails of block 2, for example, hold one banner line and one quoted line between them. Accuracy on the first set would be biased upward. We therefore measured it on 40 other emails drawn at random after the rules were final. We labelled these emails in batches of 20 and stopped once the lower end of the 95% interval for precision reached 95%. We set this threshold before labelling so that the results could not influence when we stopped. On these 40 emails the authored zone has a precision of 98.7% (236 of 239 lines; 95% interval 96.4% to 99.7%) and a recall of 97.1% (236 of 243 lines; 95% interval 94.2% to 98.8%).

4.4 Error rules

[28] Each rule targets specific categories, so that it flags the mistag and leaves correct tags in the same phrase alone ("Please let" is correctly Interactive; only "know" in "let me know" is flagged, and only as ConfidenceHigh). The rules fall into two kinds.

[29] **Fixed rules** match patterns whose tags we treat as inaccurate without coding. Block 5 applies them directly, and they stay out of the coding sample.

Rule	What it flags	Authored tags
`french_line`	Tags on lines of French text, tagged by an English-only model	871
`name`	Words inside multiword names such as COVID-19 Immunity Task Force	167
`date_month`	"May" in a date, read as the modal verb	177
`time_am_pm`	"am" after a clock time, tagged FirstPerson	150
`terms_cond`	"conditions" in "terms and conditions", tagged Contingent	23

[30] **Checked rules** depend on context, so a human sample tests each one before block 5 applies it.

Rule	What it flags	Authored tags
`domain_noun`	Data sharing nouns (agreement, consent, cohort, research, results and others)	4,040
`polite_would`	"would" or "you would" tagged Narrative	1,198
`let_know`	"know" in "let me/us know" tagged ConfidenceHigh	515

Rule	What it flags	Authored tags
`dont_hesitate`	Words in "don't hesitate to" tagged Uncertainty, Negative or ForceStressed	507
`negated_modal`	"do", "does", "did" or "can" before a negation, tagged ForceStressed or ConfidenceHedged	316
`have_a_day`	Words in "have a great day/weekend" tagged Strategic or Narrative	162
`on_my_end`	"end" in "on my/our/your end" tagged Narrative	102
`best_of`	"the best of" in "to the best of your knowledge" tagged Positive	65

[31] The `domain_noun` rule needs particular care. Domain nouns are nouns with a specific meaning in data sharing, and DocuScope gives many of them a rhetorical category. For example, DocuScope tags "agreement" as Positive, but in these emails the word often refers to a data sharing agreement.

4.5 Issues that span more than one tag

[32] Some inaccuracies occur when DocuScope splits a single name or phrase into several tags, each with its own category. For example, "COVID-19 Immunity Task Force" names an organization, but DocuScope tags "Immunity" as Positive and "Task Force" as Strategic. The inaccuracy belongs to the name as a whole, so a rule that judges each tag in isolation cannot catch it.

[33] We handle these split names and phrases in two ways. For known cases, the `name` rule searches each email for a short list of organization names and flags every tag that falls within or across one of these names. To find split names and phrases outside that list, we list runs of adjacent capitalized tags as candidate names and the 30 most frequent pairs of adjacent tags in different categories as candidate phrases.

5 The coding sample and coding tool

5.1 Rationale

[34] We code a sample of tags by hand to check that the rules themselves are accurate. The coding also measures how often DocuScope is inaccurate where no rule applies. That measure decides whether a supervised model is worth building, and the coded tags would become its first training data.

[35] Each coder reads the tagged phrase in its context and decides whether it performs the function that its category describes. The coder marks the tag `correct`, `incorrect` or `unsure`, and can add a note. The two coders work independently so that we can measure inter-coder agreement.

5.2 Stratification

[36] We stratify the sample so that each rule, domain noun and rhetorical category has enough tags to assess on its own. Within each stratum, we draw tags at random.

Stratum	Size per stratum	Strata	Tags
Each checked rule except `domain_noun`	up to 30	7	210
Each noun of `domain_noun`	up to 10	19	184
Each rhetorical category among tags that match no rule	up to 20	19	380
Total			774

[37] Each domain noun has its own stratum because nouns such as “agreement”, “access” and “results” may behave differently, and a single stratum of 30 tags across 19 nouns says almost nothing about any one of them. Each coded tag records the size of its stratum, so that block 5 can weight the results to the whole corpus. A tag that two rules match appears in the sample only once.

[38] The sample holds 774 tags, with 30 per rule, 20 per category and 10 per domain noun. These sizes are provisional, because we chose them before setting the thresholds for applying rules and rating categories. Thirty tags give each rule a confidence interval narrow enough to separate rules whose matches are nearly always inaccurate from rules whose matches are often accurate. Twenty tags give each category a coarse estimate of its inaccuracy, enough to show which

categories need closer attention. Ten tags give a first indication of how each domain noun behaves. A stratum has enough tags when its 95% confidence interval falls clearly on one side of its threshold. Once the thresholds are set, we will check each stratum against this standard and draw more tags for any stratum whose interval still spans its threshold.

5.3 The coding tool

[39] The coding tool lets two coders judge the sample quickly and independently. It is a single offline web page with the sample built in. Each card shows one tag. The card gives the category, with a short description and examples paraphrased from the DocuScope Full Dictionary documentation (DocuScope n.d.). It shows the tagged phrase with about 60 characters of context on each side, and highlights the phrase. If a rule put the tag in the sample, the card names the rule and states what the rule claims. Coders work from the keyboard. The page saves each coder's codes in the browser, and the Export CSV button writes them to a separate file for each coder. An About panel explains the purpose of the coding, how the sample is drawn, the fixed rules and how to handle the data. The page contains email excerpts, so it must stay on storage approved for the study data.

6 Preliminary comparison across tag versions

[40] Block 4 compares DocuScope tags across four versions, to see how far zoning and the error rules change the picture. It reads `rhetoric_tags.parquet`, which block 3 saves with each tag's zone and rule matches. The results are preliminary. The checked rules remain untested, and the sender groups use a provisional rule.

Version	Tags	Emails
Raw	174,765	6,055
Authored zone	142,695	5,976
Cleaned (authored zone, fixed-rule matches removed)	141,311	5,923
Upper bound (authored zone, all rule matches removed)	134,529	5,923

6.1 Category shares

[41] The overall profile holds. Narrative, Positive and Interactive remain the three largest categories, and most shares move by less than one percentage point between the raw and cleaned versions. Two categories shrink sharply. Strategic falls by 37% relative to its raw share, as "task force" (2,694 tags) leaves with

signatures and "research" drops from 3,334 tags to 512. Negative falls by 22%, as disclaimer wording leaves the authored zone. FirstPerson, Future and MetadiscourseInteractive rise by 15 to 19%, since boilerplate no longer dilutes them.

6.2 Phrases

[42] Most categories rest on a few formulas. The ten most frequent phrases cover 82% of MetadiscourseCohesive tags, and "hesitate to" accounts for about half of Uncertainty. Strategic changes most, from 60% coverage by its ten most frequent phrases in the raw version to 31% after cleaning.

[43] Misreadings survive cleaning. Positive counts "agreement" (959 tags), Interactive counts "email" (1,160), Inquiry counts "review" (473) and "assay" (280), and the most frequent ConfidenceHedged phrases, "we can" and "i can", state capability. These need rules before any further analysis.

6.3 Sender groups

[44] Block 4 assigns each email to CITF when the sender's domain matches `covid19immunitytaskforce.ca` or the sender's emails span five or more study IDs, and to study teams otherwise. CITF holds 3,477 emails from 30 senders and study teams hold 2,579 emails from 438 senders.

[45] Cleaning changes the comparison between the groups. The raw gaps in Negative, Strategic and Positive come from disclaimers, signatures and the CITF name, and they shrink or reverse once those leave. After cleaning, CITF writes more Interactive and Contingent text, and study teams write more Narrative and FirstPerson text.

[46] The grouping needs three corrections. echosign.com is an automated e-signature service and belongs in `automated_senders`. Maelstrom (349 emails) sits with CITF, and Statistics Canada (180 emails) sits with study teams, and both placements need a decision. Pooled shares can also reflect a few prolific senders, so each sender's shares need checking before the gaps are read as differences between groups.

7 Validating rules and correcting tags (planned)

[47] Block 5 is not yet written. Its planned steps:

1. **Agreement.** Merge the two coders' files, report percent agreement and Cohen's kappa overall and by sample type, and export disagreements for the two coders to settle in a consensus column. Later steps use the consensus codes.
2. **Rule validation.** For each checked rule and each domain noun, compute the share coded incorrect with a 95% confidence interval, and classify the

rule as apply, drop or refine against thresholds fixed before the results are seen.

3. **Residual accuracy.** For each rhetorical category, estimate the error rate among tags no rule matched, weighted by stratum size.
4. **Correction.** Keep authored text only, relabel tags matched by fixed rules and by confirmed checked rules, and log every change against its `msgid2`.
5. **Category reliability.** Estimate each category's accuracy after cleaning and classify categories as use, use with caution or drop, against thresholds fixed in advance.
6. **Cleaned distribution.** Rerun the category tables of `rhetoric.ipynb` block 2 on the cleaned tags and show raw and cleaned results side by side.
7. **Model gate.** Report which kept categories still carry substantial inaccuracy after cleaning. For those categories, a supervised model could judge each tag in its context from its category, phrase, surrounding words, matched rules and zone. Such a model can flag an inaccurate tag for removal but cannot supply the right category. It would also need far more coded tags than this sample holds, weighted by stratum, so building it means a further round of coding.

[48] The thresholds in steps 2, 5 and 7 remain to be set, and they must be set before the coded results are examined.

8 Planned analysis

[49] Brown and Laudenbach (2022) group the DocuScope categories into four rhetorical functions (their Table 3):

- **Negotiated Information**: CitationAuthority, PublicTerms, AcademicTerms, SyntacticComplexity, InformationReportVerbs, MetadiscourseInteractive, Citation, Contingent, InformationStates, InformationTopics, Uncertainty
- **Narrative Forecasting**: Narrative, Description, Future, Updates
- **Supportive Interaction**: Facilitate, PublicTerms, CitationHedged, Citation, Responsibility, Interactive
- **Modulated Reporting**: ConfidenceHedged, InformationChangeNegative, Uncertainty, ConfidenceHigh, FirstPerson, InformationChangePositive, Strategic, Description, InformationExposition, Negative

[50] We derive our own groups from the cleaned tags. We aggregate the rhetorical tag counts in authored text by sender and run a factor analysis, which groups the categories that the same senders use often. We call these the rhetorical groups of research data sharing emails. We name each group after its highest-scoring DocuScope categories and test the name in a manual review.

[51] The manual review covers all DocuScope tags in a sample of 200 emails. For each email, we write a topic label that names what the email is about, drawing on its InformationTopics spans, which mark the nominal subjects and objects

that signal what a text concerns. We also assign each email the rhetorical groups that fit it, and we keep a group's name only if the senders of the emails we assign it to score highest on that group.

[52] We then give part of this annotated sample to an LLM and ask it to label the remaining emails in the corpus, adding new labels where the sample has none. We check the LLM's labels against our own on the held-out part of the sample.

[53] DocuScope reads English only, so we translate the French emails and flag them. Translation changes style, so we check whether the flagged emails behave differently.

9 Reproducibility

- **Server and environment.** The CITF analysis server, Python 3.12 in `~/citf-emails-env`, built from `requirements.txt` at the repository root, with en_docusco_spacy 1.5, spaCy 3.8.18, pandas 3.1.1 and pyarrow 26.0.1.
- **Inputs.** `msgs10.csv` and `recip4.csv` in `data/`.
- **Email zoning.** Chipmunk from `webis-de/acl20-crawling-mailing-lists` at commit `ecc2c70`, with its published weights (`fasttext-model.bin` and `segmenter_hinge.epoch-11.val_loss-0.093.h5`), runs on CPU with TensorFlow 2.2 in its own environment at `~/models/email-zoning/chipmunk-env`.
- **Notebooks.** `corpus/corpus.ipynb`, `coding/rhetoric/rhetoric.ipynb`, `coding/rhetoric/tests/docuscope/docuscope.ipynb` and `corpus/tests/zoning/zoning.ipynb`, committed without their outputs.
- **Outputs** beside the notebook that writes them, which git does not track because the files contain email text: `emails_study.parquet` and `email_zones.parquet` (`corpus.ipynb` blocks 3 to 5), `tags.parquet` and `docs.spacy` (`rhetoric.ipynb` block 1), the zone check files (`zoning.ipynb`; `zone_check_` for the first set and `heldout_check_` for the second, each with `_emails.csv`, `_sample.csv`, `_tool.html` and the coder's `_coder1.csv`), `docuscope_sample_30.html` (block 2), `rhetoric_tags.parquet`, `docuscope_coding_sample.csv` and `docuscope_coding_tool.html` (block 3), and, once coded, `docuscope_coding_coder1.csv` and `docuscope_coding_coder2.csv`.
- **Randomness.** Every sample uses `random_state=42`, except the second batch of 20 emails in the second set of the zone check, which uses `random_state=62`.
- **Parameters.** Patterns, thresholds, rule lists and sample sizes sit at the top of the blocks that use them.

10 Next steps

- Decide which groups the analysis will compare, such as senders, studies or periods. Measure from the output of `rhetoric.ipynb` block 2 how much category shares differ between them.
- Set the thresholds for applying rules, rating categories and building a model from those differences, before examining any coded results.
- Set the margin, the distance a true rate can lie from its threshold and still be classified correctly with 95% confidence.
- Compute the tags needed per stratum from the thresholds and margin, and update `n_per_rule`, `n_per_category` and `n_per_noun` in block 3.
- Add a top-up draw to block 3 that samples more tags for chosen strata without repeating tags already drawn. Make the coding tool load the new tags alongside existing codes.
- Test the `name` rule in the coding sample. The rule currently counts as accurate without testing, flags tags that only partly overlap a name, ignores case and covers only seven names.
- Write content patterns that exclude automated messages sent from personal addresses, such as out-of-office replies and ticket confirmations.
- Have two coders code the sample in `docuscope_coding_tool.html`.
- Add echosign.com to `automated_senders`, write rules for the misreadings block 4 found, and check each sender's shares (Section 6).
- Write and run block 5 (Section 7).

References

Bevendorff, Janek, Khalid Al Khatib, Martin Potthast, and Benno Stein. 2020. "Crawling and Preprocessing Mailing Lists at Scale for Dialog Analysis." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 1151–58. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.108.

Brown, David West, and Michael Laudenbach. 2022. "Stylistic Variation in Email." *Register Studies* 4 (1): 1–29. https://doi.org/10.1075/rs.20023.bro.

DocuScope. n.d. "DocuScope Tags (Full Dictionary)." Accessed November 12, 2026. https://browndw.github.io/docuscope-docs/tagsets/docuscope-fd-tags.html.

Jardim, Bruno, Ricardo Rei, and Mariana S. C. Almeida. 2021. "Multilingual Email Zoning." In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, 88–95. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.eacl-srw.13.

Lampert, Andrew, Robert Dale, and Cécile Paris. 2009. "Segmenting Email Message Text into Zones." In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, 919–28. Singapore: Association for Computational Linguistics. https://aclanthology.org/D09-1096/.

Repke, Tim, and Ralf Krestel. 2018. "Bringing Back Structure to Free Text Email Conversations with Recurrent Neural Networks." In *Advances in Information Retrieval: 40th European Conference on IR Research*, 114–26. Lecture Notes in Computer Science. Cham: Springer. https://doi.org/10.1007/978-3-319-76941-7_9.