Evaluating large language-vision models on geographic language understanding

Anonymous ACL submission

Abstract

Geographic language understanding (GLU) tasks ask models to map from text to maps. Geographical complex description parsing (GCDP) is a GLU task where models must assign sets of map coordinates to text that goes beyond a single named location, such as "*...between the towns of Adrano and S. Maria di Licodia, 32 kilometres northwest of Catania*". In GCDP, the input is both a text and a set of reference geometries for known places in the text (e.g., *Adrano*, *S. Maria di Licodia*, *Catania*), and the output is the geometry of the location described. In this paper, we convert a GCDP corpus into an `image + text` → `image` benchmark to evaluate recent large langugage-vision models on such complex task. The models show weak performance, with analysis showing a lack of understanding of even simpler tasks like recognizing regions by color.

1 Introduction

The goal of geographic language understanding (GLU) is to develop models that can map from descriptions of locations in text to the corresponding locations on a map. A commonly studied GLU task is geoparsing, which asks models to map mentions of locations in text to their geographical geometries, formed of sets of coordinates, typically by linking mentions to entries in a toponym database like GeoNames[1] (Gritta et al., 2018; Zhang and Bethard, 2023). A more complex GLU task is geographical complex description parsing (GCDP), where the input is a description of a geographical region and a list of reference geometries (sets of coordinates), and the goal is to predict the geometry of the region described (Laparra and Bethard, 2020). For example, the text *"a town and comune in the Metropolitan City of Catania, Sicily, southern Italy... located between the towns of Adrano and S.Maria di Licodia, 32 kilometres (20 mi) northwest of Catania."* describes a location that is not explicitly named. The goal is to approximate the geometry of the location using the description as input along with the geometries of the reference toponyms: *Catania*, *Sicily*, *Italy*, *Adrano*, etc.

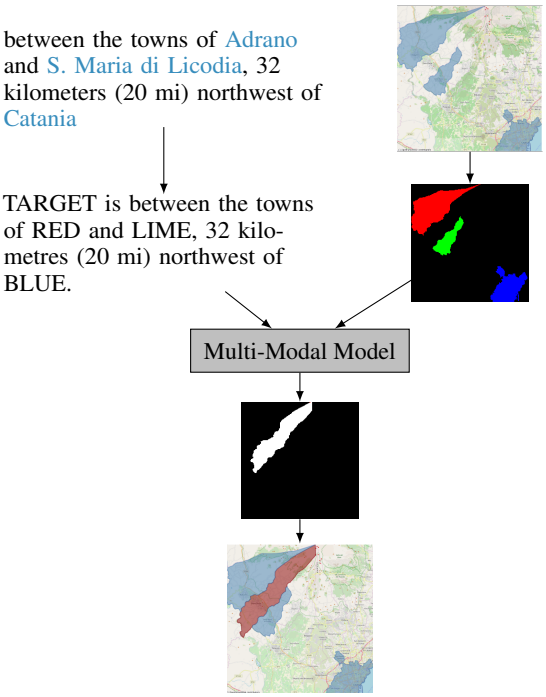

Figure 1: Framing geographic geometry prediction as a multi-modal `image + text` → `image` problem. The reference toponyms are the blue phrases. The reference geometries are the blue map regions. The target geometry is the red map region. The color-on-black images are our approach to translating between maps and images.

To solve this task, Laparra and Bethard (2020) propose a grammar-based baseline that parses these descriptions into a composition of spatial operators, deterministic functions whose composition yields the target geometry. This baseline achieves 0.221 F1, leaving substantial room for improvement with machine learning methods. However, a major challenge in applying machine learning to this task is that the reference geometries, be they polygons or linestrings, are represented by an undefined number of coordinates, ranging from just a few to over

[1] http://www.geonames.org

a million. Although methods exist to obtain embedding representations of the geometries for their use in machine learning (Mai et al., 2022), to date there is no clear way to get current language models to output such geometries.

We consider an alternative to predicting coordinate sets: convert geometries to bitmap images and apply multi-modal language-vision models (LVM), as shown in Figure 1. Our contributions are [2]:

- We propose a strategy to convert GCDP into an `image + text` $\rightarrow$ `image` problem and evaluate two LVMs designed to work in this setting.
- Due to the high difficulty of the task, We develop 4 variants of the dataset, each designed to be simpler than the GCDP task and individually analyze a different required skill to solve the task.
- We find that although the current models show some ability to solve the task, their failures stem from lack of understanding of simpler tasks like recognizing regions by color.

2 Datasets

We use the Laparra and Bethard (2020) GCDP corpus, derived semi-automatically from Wikipedia and Openstreemap, that contains a training set of 360,187 uncurated examples and a test set of 1,000 manually curated examples. We select 67,293 and 1,000 examples for training and development respectively where all the reference locations in the description has a geometry associated (see Appendix A). We use the same test set as Laparra and Bethard (2020) to allow results comparison.

In the following sections, we first introduce how we translate the GCDP problem into an `image + text` $\rightarrow$ `image` problem. Then we introduce our proposed dataset variants, shown in Figure 2, that allow the study of different capabilities of LVMs. See Appendix A for dataset generation details.

2.1 Image-based Dataset

Obtaining an image-based dataset from GCDP data requires decisions of which part of the world map to show in the image and how to link the reference geometries in the image to the reference toponyms in the text. Our strategy is as follows.

Decide boundary: To create an image, we must first select a small region of the map, as using the entire map would result in most locations being smaller than a single pixel. A good region for GCDP should completely include the target geometry, represent such geometry with a sufficient number of pixels and include at least a portion of every reference geometry. However, the target geometry is not known at prediction time and thus should not be used when selecting the boundary. We thus use a heuristic: set the boundary to 100 km in each cardinal direction from the geometric median of the centroids of the reference geometries.

Link reference geometries and toponyms: Given the boundary, we create a pixel grid with $N \times N$ pixels where each corner corresponds to each coordinate of the boundary. For the input image, we overlay the grid with the reference geometries, assigning a different color to each of them, calculating the average of the colors in RGB space when geometries overlap. For the output image, we overlay the grid with only the target geometry in white. To link the reference geometries in the image with the reference toponyms in the text, we replace each toponym in the text with the name of the color used to represent the corresponding geometry. The middle of Figure 1 visualizes this and the preceding step.

2.2 Oracle Boundary

The method to generate the boundary shown in Section 2.1 may result in a wide boundaries where the target geometry is represented with just a few pixels. To better understand how this size affects LVMs, we develop an oracle version of the image-based dataset where we generate the images looking for the minimum boundary that covers the target geometry (hence an oracle) and at least a portion of all the reference geometries. We start initially from the envelope covering the target geometry and extend it until it touches at least one point of all reference geometries. We add 10 kilometers in all 4 cardinal directions to ensure that the images include a portion of all geometries.

2.3 Colored Region Identification

For the `image + text` $\rightarrow$ `image` approach to work, it is essential that the models are able both to relate textual mentions of colors to those colors in the images, and to differentiate objects of a given color from the other objects in the image. However, this is not possible to analyze in detail in the GCDP dataset, due to the complexity of the task. Therefore, we generate two datasets where the text simply states the color of the target in the input

[2] Code and data will be available.

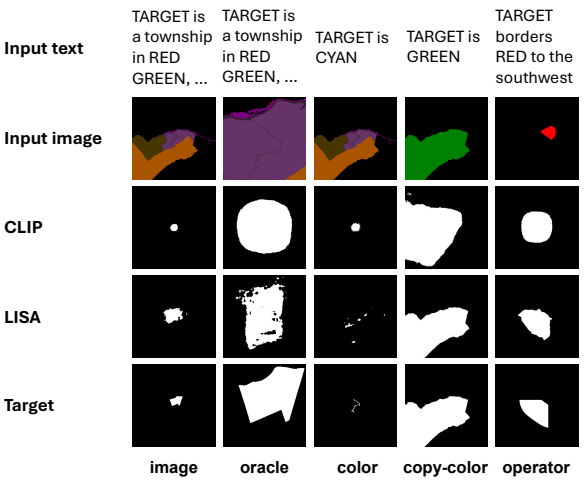

Figure 2: Examples from the five datasets of input text, input image, both LVM predictions, and target image.

image, e.g., "*TARGET is RED*".

Copy Color The input image contains only the geometry of the target image, and the output image is the same as the image dataset. The purpose of this dataset is to test whether the models are able to recognize a region in an image based on color.

Color The input image contains the target geometry and all reference geometries, and the output image is the same as image dataset. The purpose of this dataset is to test whether the models are able to differentiate objects of the color mentioned in the text from objects of other colors.

2.4 Single Operator Understanding

The `image + text` $\rightarrow$ `image` approach requires the models to be able to interpret the spatial relationships described in the text and to calculate the corresponding operations to obtain the target geometry. To better understand to what extent these models are able to achieve this, it would be necessary to analyze if they can interpret a single spatial relation (e.g. "between") in isolation. The dataset by Laparra and Bethard (2020) does not allow us to perform this analysis as its input descriptions correspond to compositions of spatial relations, and its output is only the final composed geometry, not the outputs of any of the individual relations.

Thus, we generate a synthetic dataset where the descriptions only include a single spatial relation using the grammar defined by Laparra and Bethard (2020) and their deterministic implementation of the spatial operators. A description in this dataset looks like "*TARGET is 50 km Southwest of RED*".

3 Models

We test two different LVMs on our datasets using 224×224 images in all cases:

CLIP (Radford et al., 2021) is a transformer-based LVM pretrained on an image and text pairing task. We evaluate this model because of its popularity and success on several multi-modal NLP tasks. To predict the target region, we use the CLIP model as an encoder to extract text and image features of the input. The fused text and visual features are fed to a decoder to generate the target region. See Appendix B for model implementation details.

LISA (Lai et al., 2023) is a transformer-based LVM trained for a reasoning segmentation task, in which the model needs to generate a segmentation mask by reasoning from the input text and image. We evaluate this model due to the similarity of its pre-trained task and the image-based version of GCDP. We use the same setting as the original LISA paper and finetune the model on our datasets.

3.1 Training Objectives

For the CLIP model, we use a combination of the DICE loss and per-pixel binary cross-entropy (BCE) loss as the training objective:

$$\mathcal{L}_{\text{CLIP}} = \lambda_1 \mathcal{L}_{\text{DICE}} + \lambda_2 \mathcal{L}_{\text{BCE}}$$

DICE loss is a commonly used loss function in image segmentation tasks, which is defined as:

$$\mathcal{L}_{\text{DICE}} = 1 - \sum_{i,j} \frac{2 p_{ij} y_{ij} + 1}{p_{ij} + y_{ij} + 1}$$

where $p_{ij} \in [0, 1]$ is the i-th row, j-th column element value of the prediction image and $y_{ij} \in \{0, 1\}$ is the i-th row, j-th column element value of the ground truth image. The DICE loss is useful for unbalanced datasets such as ours where the target region makes up a small proportion of the output image, and we want to prefer predicting the target region to predicting every pixel as negative. In our experiments, λ_1 are λ_2 are set to 1.

For the LISA model, as the language encoder generates a sentence which should contain a special token representing the prediction, we follow the original paper and add an extra text generation loss besides the DICE loss and the BCE loss:

$$\mathcal{L}_{\text{LISA}} = \lambda_1 \mathcal{L}_{\text{DICE}} + \lambda_2 \mathcal{L}_{\text{BCE}} + \lambda_3 \mathcal{L}_{\text{txt}}$$

$\mathcal{L}_{\text{txt}}$ is the cross-entropy loss between the language model predicted word and the teacher-forcing label. In our experiments, λ_1, λ_2 are λ_3 are set to 1.

	Strict			Relaxed		
model	P	R	F_1	P	R	F_1
GRAMMAR	17.2	31.0	22.1	21.3	27.6	24.0
CLIP	7.2	27.0	11.3	13.4	26.1	17.7
LISA	9.1	38.7	14.7	15.5	36.7	21.8

Table 1: Comparison of LISA and CLIP using the IMAGE dataset with the grammar-based baseline proposed by Laparra and Bethard (2020). The figures are based on the GCDP metrics. More details in Appendix C

	CLIP			LISA		
dataset	P	R	F1	P	R	F1
IMAGE	12.7	34.1	18.5	15.9	31.2	21.1
ORACLE	21.0	20.4	20.7	35.9	42.6	39.0
COLOR	38.2	56.1	45.5	56.1	61.5	58.7
COPY COLOR	43.7	59.4	50.3	73.5	82.1	77.5
OPERATOR	23.9	26.1	25.0	89.7	89.8	89.8

Table 2: Performance of CLIP and LISA model on the five datasets using the image-based metrics.

4 Metrics

For comparison with the grammar-based baseline by Laparra and Bethard (2020), we use the metrics they proposed that are based on the overlap between the predicted and target geometries. In the *strict* version, the overlap is calculated with the original target geometry while the *relaxed* version uses its envelope, i.e., the rectangle that encloses the geometry. For this evaluation, the predicted images must be translated back to a set of coordinates.

For the analysis on the different datasets, we apply the following image-based metrics. Let the area of the target region be S_t, the area of the predicted region be S_p, the area of $S_t \cap S_p$ be S_I, we evaluate the performance of the models using the per pixel precision P, recall R, and $F1$ score.

$$P = \frac{S_I}{S_t} \qquad R = \frac{S_I}{S_p} \qquad F1 = \frac{2PR}{P+R}$$

5 Results

Table 1 shows that LISA outperforms CLIPS on GCDP and is comparable to the baseline proposed by Laparra and Bethard (2020) in the relaxed evaluation. It also obtains a higher recall. However, LVMs seem still to be far from a manually constructed grammar for GCDP. The results in Table 2 provide some insight into why these models have difficulties to solve GCDP. Figure 2 shows example predictions of the two models. We observe that:

The size of the target regions in the data significantly influences the model performance. F1 score of the CLIP model increases 2.2 points and the LISA performance nearly doubles when using the ORACLE boundaries where the target regions are a larger portion of the image. Both models also perform better in the COPY COLOR, COLOR, and OPERATOR datasets where the target region area is also generally larger. This suggests that in real-world settings where an oracle boundary is not available, finding a good boundary is key to improving model performance.

Segmentation-based pre-training helps to understand spatial relations in text. The results in Table 2 show a huge gap between CLIP and LISA on the OPERATOR dataset. This means that LISA understands better spatial relations in text and is better able to reason on the image accordingly. This also contributes to LISA's better performance on IMAGE and ORACLE datasets.

Segmentation-based pre-training helps to capture the shape of the target regions better. As shown in Figure 2, CLIP tends to generate mostly circle-like shapes in the middle of the image. While this guarantees some recall of the prediction, the overall precision of CLIP is low. LISA captures the shape of geometries better. LISA can get a near-perfect target shape in the COPY COLOR dataset and a very close guess when predicting the result on OPERATOR. This is also verified by the high performance of LISA on these two datasets.

Colors are more difficult to understand than shapes. Understanding colors is crucial for the model to capture the relationship between the input text and image. The task not only requires the model to relate the color words to colors in the image, but also requires the model to understand how different colors mix when there are overlapping regions. Our results show that when this kind of color understanding is required, models tends to perform poorly. This is indicated by the low performance of both models on the COLOR, ORACLE, and IMAGE datasets. This suggests that more work is needed to infuse color knowledge into LVMs, and that it may be worth exploring ways of representing geometries in images that do not rely on color.

It is difficult for the models to predict linestrings such as rivers or roads. These objects are usually in a long and complex shape but small in total area. The LVMs have a hard time generating such shapes. An example of this is the COLOR column of Figure 2, where the target is a river, and neither model makes an accurate prediction.

Limitations

The aim of the paper is to give an insight into the potential and limitations of multi-modal language-vision models for GCDP. Two of these models have been selected based on their success in various NLP tasks or on the similarity of their pre-initialization task with GCDP. However, the study does not cover the full range of existing multi-modal language-vision models.

Intended Use and Ethical Concerns

The data and models we developed in this paper is intended to be used on GCDP tasks. We do not foresee any immediate ethical concerns of our work. However, we acknowledge that as we use LVMs in our experiments, the models may generate unexpected images if not properly used by an user or not used on this task.

References

Milan Gritta, Mohammad Taher Pilehvar, Nut Limsopatham, and Nigel Collier. 2018. Whatś missing in geographical parsing? *Language Resources and Evaluation*, 52(2):603–623.

Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. 2023. Lisa: Reasoning segmentation via large language model. *arXiv preprint arXiv:2308.00692*.

Egoitz Laparra and Steven Bethard. 2020. A dataset and evaluation framework for complex geographical description parsing. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 936–948, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Gengchen Mai, Krzysztof Janowicz, Yingjie Hu, Song Gao, Bo Yan, Rui Zhu, Ling Cai, and Ni Lao. 2022. A review of location encoding for geoai: methods and applications. *International Journal of Geographical Information Science*, 36(4):639–673.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

Zeyu Zhang and Steven Bethard. 2023. Improving toponym resolution with better candidate generation, transformer-based reranking, and two-stage resolution. In *Proceedings of the 12th Joint Conference on Lexical and Computational Semantics (*SEM 2023)*, pages 48–60, Toronto, Canada. Association for Computational Linguistics.

A Dataset Generation

The corpus by Laparra and Bethard (2020) contains 360,187 uncurated examples and a test set of 1,000 manually curated examples. In our experiments, we use the same test set. For training, we use the uncurated portion of the corpus, however this portion does not guarantee that all the locations in the descriptions are mapped to their corresponding geometry. We run named-entity recognition on the uncurated examples to obtain all the location mentions, and check if the recognized locations are linked to a geometry. We keep only those descriptions that have all the recognized locations linked. As a result, we obtain 68,293 examples from which we use 67,293 as training set and 1,000 as development set.

A.1 Image-based Dataset

Below we detail the steps we follow for the conversion of this dataset:

Generate boundary: From the reference geometries, we first discard the geometry with the largest area in order to avoid boundaries that are too wide for the target geometry.[3] Then, we calculate the geometric median[4] of the centroids of the remaining reference geometries. We set the boundary to 100 km in each cardinal direction from this centroid and obtain the coordinates of each of the 4 corners of this boundary

The advantage of generating the same width for all boundaries is that in the resulting images the same spatial distance will be represented with the same number of pixels, i.e., there will be a proportional relation between the distances mentioned in the text and the distances in the images for all cases. For example, if a description mentions a distance of "100 km" between two locations and it is represented as 40 pixels in the resulting image, a mention of "50 km" in another description will be represented with 20 pixels in the corresponding image.

Generate images: Once we calculate the boundary for the images, we apply postgis' *st_asraster*[5] function to obtain a bitmap representation of the geometries. The function creates a pixel grid with $N \times N$ pixels where each corner corresponds to each coordinate of the boundary. The function overlays this grid on a geometry and calculates if each pixel intersects with the geometry, assigning 1 if true and 0 otherwise. After obtaining a bitmap image (raster) for each geometry, each pixel grid is translated into a RGB format, using a different color for each geometry. Then, all these RGB pixel grids are joined in a single image. Where 2 or more colors overlap in one pixel, we calculate the average. E.g. for a pixel where (255, 0, 0) and (0, 255, 0) overlap, we assign (128, 128, 0) in the final image. In the case of the target image, only one geometry will be part of it and we use the white color (255, 255, 255) to represent it.

Update the descriptions: The last step consists of replacing all the mentions of each location in the description with the name of the color used in the previous step to represent the corresponding geometry.

A.2 Oracle Boundary

To find the oracle boundary, we start initially from the envelope covering the target geometry and extend it until it touches at least one point of all reference geometries. Finally, we extend the boundary 10 kilometers in all 4 cardinal directions to ensure that the images include a portion of all geometries. Once the boundary is obtained, the reference and target images are generated as described in 2.1.

Unlike the dataset described in 2.1 where all images correspond to the same spatial extent, the boundary of the images in this version may cover different extents in each case, which does not guarantee the correspondence between the distance units described in the text with the distance in pixels of the images. To solve this problem, we automatically modify the spatial units mentioned in the text by scaling them appropriately. First, for each case, we calculate the ratio between the number of pixels of the width of the images and the width in kilometers of the boundary. Then we identify by a simple regular expression all mentions of distance units, e.g. "100 KM", extract the quantity and multiply it by the ratio calculated in the previous step. Finally, we modify the text with the result of this calculation rounded to the nearest integer.

A.3 Colored Region Identification

To generate each example of these datasets, we assign a color to each reference geometry as explained in Section 2.1. We then randomly select

[3] The geometry is discarded only to calculate the boundary but it is included in the resulting image.

[4] The geometric median is more robust to outliers than the centoid

[5] https://postgis.net/docs/RT_ST_AsRaster.html

	Strict					Relaxed					Coverage
model	P	R	F_1	P_{x2}	R_{x2}	P	R	F_1	P_{x2}	R_{x2}	%
GRAMMAR	0.172	0.310	0.221	0.272	0.381	0.213	0.276	0.240	0.358	0.365	52.8%
CLIP	0.072	0.270	0.113	0.169	0.456	0.134	0.261	0.177	0.288	0.451	100%
LISA	0.091	0.387	0.147	0.205	0.566	0.155	0.367	0.218	0.315	0.560	87.5%

Table 3: Comparison of LISA and CLIP with the grammar-base baseline proposed by Laparra and Bethard (2020).

one of the reference geometries and obtain a target image containing only that geometry. The description in this case will be simply "TARGET is COLOR", where COLOR corresponds to the color assigned to the selected geometry. Finally, in each of the datasets, we follow a different strategy to generate the reference image:

COPY COLOR The reference image contains only the geometry selected for the target image.

COLOR The reference image contains all reference geometries including the one selected for the target image.

A.4 Single Operator Understanding

Each example in this dataset is generated following the next steps:

1. One of the operators implemented by (Laparra and Bethard, 2020) is randomly selected and the values of the corresponding arguments are also randomly obtained. For example, if the operator takes a cardinality as argument, its values selected among the possible values None, North, Northeast, and so on.
2. Select a possible pattern defined in the grammar for the operator selected in the previous point and complete it with the selected values. For example, a possible pattern could be "TARGET is [Distance] [Unit] [Cardinal] of [Reference]" which could be completed as "TARGET is 50 km Southwest of REFERENCE".
3. We generate the necessary reference geometries randomly. For instance, we should generate a geometry for the REFERENCE in the previous example.
4. Applying the corresponding operator with the values of the arguments obtained in step 1 and the references in step 3, we obtain a new geometry that would correspond to the TARGET of the description generated in step 2.

After this process, the reference and target images are generated as explained in Section 2.1, assigning a random color to the reference geometry and white to the target. The description is also updated accordingly.

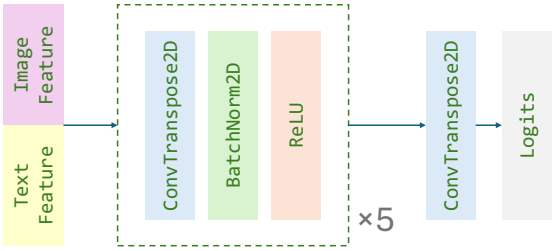

Figure 3: CLIP decoder takes the concatenated text and image feature vectors as input and construct the output image with 2D transposed convolution layers.

B Model Implementation and Training Details

B.1 CLIP

The original CLIP model limits the maximum input text token length to be 77. To input longer texts in our dataset, we replace the original input projection layer (length of 77) with a 512-long projection layer. We initialize the first 77 elements of this layer with the pretrained CLIP weights and initialize the remaining of the layer randomly.

The structure of the CLIP decoder is shown in Figure 3. We concatenated the output the CLIP text encoder and image encoder as the input of the decoder. The decoder is a stack of tranposed convolution layers (also known as deconvolution layers).

In our experiments, we set the learning rate to 0.0002 and finetune the model for 5 epochs. We used one Nvidia A100 GPU for finetuning, and finetuning each task takes about 3 hours.

B.2 LISA

We follow the setting of the original LISA work, instead that we set the learning rate to 0.00003. We used the pretrained LISA-7B-v1 in our experiments and finetune the model for 1 epoch. We used 4 Nvidia A100 GPUs for finetuning, and finetuning each task takes about 13 hours.

C GCDP Evaluation

Table 3 shows the performance of CLIP and LISA on GCDP. The table shows all the metrics proposed by Laparra and Bethard (2020).