Our vision is therefore to extract procedural knowledge (PK) from those documents, and to build a knowledge graph (KG) according to an ontology. This KG can then be used by different downstream applications to facilitate the access and use of such procedures by human operators.

Examples of such applications could be (KG-empowered) search applications or intelligent assistant: we expect the users to feel the need to be helped to find a specific procedure (or a part thereof) and to be guided step-by-step in its execution, for example by being informed about the action they have to perform (e.g., turning off a switch), the equipment they may need to use (e.g., wearing protective gloves) or the time it may take to perform a specific step (e.g., approximately 15 minutes).

In order to fulfill such requirements, the procedural KG extracted from text should (1) preserve the intended meaning of the original document and (2) contain enough information to guide a user in correctly executing the procedure.

The extracted KG should therefore be evaluated, respectively, on the basis of its quality and usefulness (cf. Section 7). Based on this scenario, we define a simple ontology and we identify a general-purpose dataset to be used in our LLM-powered PK extraction and KG building experiments.

Ontology. We reuse existing ontologies when applicable, namely: P-Plan [14] and K-Hub [42], that address plans and related concepts, FRAPO, one of the SPAR ontologies3, and the Time Ontology4, while creating a few new classes

and properties when needed (po:). Specifically, as depicted in Figure 1, a proce
dure is represented by the class p-plan:Plan and is linked to its p-plan:Steps
(po:hasStep), which are sequentially ordered (property p-plan:precededBy
and its inverse khub-proc:nextStep). Each step is then linked to its frapo:E
quipment, if any, via the property frapo:usesEquipment, and with the action(s)
to be performed while executing it (po:hasAction), along with the direct object
of the action (po:hasDirectObjectOfAction). The information about the time
needed for executing the step is represented by the class time:TemporalEntity.
Input procedures. For our experiments, we use as a reference dataset Wiki
How, one of the largest online databases of PK, which includes how-to articles
on multiple domains. We reuse the JSON dataset built by [52], crawled from
the WikiHow website5, and focus on atomic procedures, that do not include
methods/parts as sub-procedures. Out of each selected procedure, we build an
unformatted text (also partially removing punctuation), by concatenating (i)
procedure title, representing the overall goal, (ii) general procedure description,
(iii) headline of each step, (iv) description of each step. Thus, we also include
in the text irrelevant sentences, which are supposed to be discarded during the
extraction phase. In total, we used four procedures randomly selected from Wik
iHow, one as working example for our prompt engineering phase, and three for
our method replication and human assessment. Such procedures are

diverse with
 respect to both complexity and topic: how to clean a computer monitor (working
 example), how to fix a rubbing door, how to make honey glazed parsnips, and how
 to plant a bare root tree.